EX LIBRIS
UNIVERSITATIS
ALBERTENSIS

The Bruce Peel
Special Collections
Library

Digitized by the Internet Archive
in 2025 with funding from
University of Alberta Library

https://archive.org/details/0162017486604

University of Alberta

Library Release Form

Name of Author: Jinhui Shen

Title of Thesis: Position-Based Routing With A Power-Aware Weighted Forwarding Function In MANETs

Degree: Master of Science

Year this Degree Granted: 2003

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as hereinbefore provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.

University of Alberta

POSITION-BASED ROUTING WITH A POWER-AWARE WEIGHTED FORWARDING FUNCTION IN MANETS

by

Jinhui Shen

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of **Master of Science**.

Department of Computing Science

Edmonton, Alberta
Fall 2003

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **Position-based Routing With A Power-Aware Weighted Forwarding Function In MANETs** submitted by Jinhui Shen in partial fulfillment of the requirements for the degree of **Master of Science**.

Abstract

In this thesis, we present a novel position-based routing protocol with a Power-Aware Weighted Forwarding function, named PAWF. With PAWF, the energy consumption and energy residual values of mobile nodes are asymmetrically combined with forwarding achievement and used for hop by hop routing. PAWF routing requires only the local information of strict neighbors and the destination. In order to keep the accuracy of neighbor tables, MAC layer failure feedback and piggybacking strategies are used. An existing planar graph face-crossing algorithm is also used by PAWF to handle the local maxima problem of greedy position-based routing.

With the power-awareness of PAWF, the over-usage of mobile nodes is avoided and the lifetime of the network is increased greatly. Based on simulation results, PAWF is shown to be able to provide global energy-sufficient near-optimal paths by making locally optimal choices, which offers the possibility to implement further QoS routing in MANETS.

Acknowledgements

I would like first to thank my research advisor, Janelle Harms. Her guidance has been an invaluable resource throughout all stages of this thesis. My discussion with her always leads to new insights in depth and width. Moreover, I also learnt from her that research requires a long-term thinking, systematic organization and strict confidence, which will be benefitting me more in my future work.

I am also very appreciated to the help of Professor Ioanis Nikolaidis, who made himself available for a lot of discussion. His wisdom and instructive advices are important to the accomplishment of this thesis.

My wife, HongBo, have given me love and encouragement all the time. I thank her for the cares and sacrifices. Without her, this thesis would not have been possible.

Tian He and Kui Wu are two who deserve many thanks. I have shared much creative discussion with them. Kui Wu gave me very meaningful feedbacks about protocol modelling and system design. Tian He offered me many insights about mobile Ad Hoc networks and position-based routing, especially the simulation design over GlomoSim. Special appreciations to Kyungtae Woo for discussing about power-aware routing and generously offering the source code of the LEAR protocol. I am especially grateful for the advices from Yuan Sha and Qiong Xie. I have enjoyed many chats with them that have enriched my view of energy consumption modelling and telecommunication networks. Many thanks to Ioannis Batsiolas for explaining propagation model and energy consumption in mobile networks. Thanks also to Tommy Chu for the analysis of the random waypoint mobility model and to Kun Bai for the help of the code debug.

Finally, I would like to thank my friends and classmates, Yuxi Li, YanXia Jia and Qiang Ye, for the helps in my study and research during the passing two years.

Contents

1 Introduction **1**
- 1.1 Motivations . . . 1
- 1.2 Contributions: Power-Aware Weighted Forwarding (PAWF) Routing Protocol . . . 2
- 1.3 Layout of the Thesis . . . 4

2 Related Work **6**
- 2.1 Ad Hoc Networks . . . 6
- 2.2 Regular Routing in MANETs . . . 8
  - 2.2.1 Topology-based Routing . . . 8
  - 2.2.2 Position-based Routing . . . 10
- 2.3 Power-Aware Routing in MANETs . . . 12
  - 2.3.1 Energy Saving . . . 12
  - 2.3.2 Energy Balance of Nodes . . . 16
- 2.4 Chapter Summary . . . 17

3 Power-Aware Weighted Forwarding (PAWF) Protocol **18**
- 3.1 Research Goal . . . 18
- 3.2 Assumptions and Specifications . . . 20
- 3.3 Position-based Routing . . . 22
  - 3.3.1 Physical Environment of MANETs . . . 22
  - 3.3.2 Position-based Routing . . . 23
- 3.4 Power-Aware Weighted Forwarding (PAWF) Routing . . . 26
  - 3.4.1 Greedy Forwarding with An Asymmetrical Exponential Weighted Forwarding Function . . . 27
  - 3.4.2 Beacon Method to Exchange Neighbor Information . . . 33
  - 3.4.3 Adaptation to High Mobility and Topology Change . . . 34
  - 3.4.4 Recovery of Greedy Forwarding Failure . . . 35
  - 3.4.5 The Full Routing Process of PAWF . . . 41
- 3.5 Chapter Summary . . . 48

4 Simulations and Experiments **51**
- 4.1 Simulation Environment . . . 51
  - 4.1.1 Physical Layer and MAC Layer Settings . . . 51

4.1.2 The Internet Protocol (IP) . . . 53
4.1.3 Workload Model . . . 54
4.1.4 Mobility Model . . . 54
4.1.5 Energy Consumption Model . . . 55
4.1.6 Summary . . . 55
4.2 Simulation Metrics . . . 55
4.3 Experiment Design . . . 57
4.4 The Implementation of AODV, LEAR and GPSR . . . 58
4.4.1 The Implementation of AODV . . . 58
4.4.2 The Implementation of LEAR . . . 59
4.4.3 The Implementation of GPSR . . . 64
4.5 Analysis of Experimental Results . . . 64
4.5.1 Routing Performance with Enough Energy . . . 64
4.5.2 Routing Performance with Insufficient Energy . . . 76
4.5.3 Routing Performance with Inefficient Energy and Battery Reloading 82
4.6 Chapter Summary . . . 85

5 Discussion **87**
5.1 The PAWF Properties with Different Network Characteristics . . . 87
5.2 PAWF vs. Energy Consumption . . . 91
5.3 The Effect of Location Services . . . 93
5.4 Chapter Summary . . . 95

6 Conclusions and Future Work **97**
6.1 Conclusions . . . 97
6.2 Future Directions for Work . . . 99

Bibliography **101**

List of Figures

3.1 Local maxima of greedy forwarding 25
3.2 Pseudocode for position-based routing process 26
3.3 Pawf Model 27
3.4 Unnecessary relay with a linear weighted forwarding function 28
3.5 Asymmetrical exponential weighted forwarding function 30
3.6 Pseudocode for greedy power-aware weighted forwarding 31
3.7 Structure of a beacon message 32
3.8 Structure of a neighbor table entry 32
3.9 Pseudocode for beacon receiving and neighbor table update 33
3.10 The right-hand rule traverses the interior of the square. B receives data from A first and the traversing path is: $A \rightarrow B \rightarrow C \rightarrow D \rightarrow A$ 36
3.11 The RNG graph 37
3.12 planar graph and planar straight line graph 38
3.13 Face-crossing traversal: $\overline{SD}$ crosses the face 1, 2 and 3 progressively from S to D, the arrows show the path chosen by the rule of face-crossing 39
3.14 Pseudocode for the generation of RNG 40
3.15 Packet header of PAWF for the recovery process 42
3.16 Pseudocode for the full routing process of PAWF 43
3.17 PAWF routing process 45
3.18 The multi-layer design of PAWF and the APIs 47
3.19 Nodes with PAWF protocol 49

4.1 Packet delivery ratio: 300 CBRs, CBR interval = 0.2s/packet, full battery initially, simulation time = 1500s 65
4.2 Routing overhead on different mobility levels: 300 CBRs, CBR interval = 0.2s/packet, full battery initially, simulation time = 1500s 65
4.3 Average end-to-end delay per packet : 300 CBRs, CBR interval = 0.2s/-packet, full battery initially, simulation time = 1500s 66
4.4 Average energy consumed per packet : 300 CBRs, CBR interval = 0.2s/-packet, full battery initially, simulation time = 1500s 66
4.5 COV of energy consumed for nodes: 300 CBRs, CBR interval = 0.2s/-packet, full battery initially, simulation time = 1500s 67
4.6 COV of the workload for nodes : 300 CBRs, CBR interval = 0.2s/packet, full battery initially, simulation time = 1500s 67

4.7 Average number of hops per packet : 300 CBRs, CBR interval = 0.2s/-packet, full battery initially, simulation time = 1500s 68
4.8 Routing overhead on different network workloads: 300 CBRs, Pause = 0s, full battery initially, simulation time = 1500s 68
4.9 Energy consumption for transmitting and receiving routing messages on heavy workload : 300 CBRs, CBR interval = 0.01 s/packet, pause = 0s, full battery initially, simulation time = 1500s 69
4.10 Lifetime of the network under different movement patterns: 300 CBRs, CBR interval = 0.2s/packet, limited energy initially, simulation time = 1500s 74
4.11 Number of packets delivered during the network lifetime: 300 CBRs, CBR interval = 0.2s/packet, limited energy initially, simulation time = 1500s . . 74
4.12 Packet delivery ratio during the simulation time: 300 CBRs, CBR interval = 0.2s/packet, limited energy initially, simulation time = 1500s 75
4.13 Lifetime of the network on different network workloads: 300 CBRs, pause = 0s, limited energy initially, simulation time = 1500s 75
4.14 Lifetime of the network under different movement patterns: 300 CBRs, CBR interval = 0.2s/packet, limited energy initially, simulation time = 1500s 78
4.15 Packet delivery ratio during the simulation time: 300 CBRs, CBR interval = 0.2s/packet, limited energy initially, simulation time = 1500s 78
4.16 Routing overhead during the simulation time: 300 CBRs, CBR interval = 0.2s/packet, limited energy initially, simulation time = 1500s 79
4.17 Average end-to-end delay per packet during the simulation time: 300 CBRs, CBR interval = 0.2s/packet, limited energy initially, simulation time = 1500s 79
4.18 Number of nodes being reloaded: 600 CBRs, CBR interval = 0.2s/packet, limited energy initially, battery warning threshold = 5000mWsec, battery reloading period = 1-10mins, simulation time = 3000s 83
4.19 Number of nodes being reloaded when alive: 600 CBRs, CBR interval = 0.2s/packet, limited energy initially, battery warning threshold = 5000mWsec, battery reloading period = 1-10mins, simulation time = 3000s 83
4.20 Packet Delivery Ratio: 600 CBRs, CBR interval = 0.2s/packet, limited energy initially, battery warning threshold = 5000mWsec, battery reloading period = 1-10mins, simulation time = 3000s 84

5.1 The routing performance of PAWF with/without MAC layer failure feedback: 300 CBRs, CBR interval = 0.2s/packet, pause time = 0s, simulation time = 1500s 88
5.2 Inner-face problem: endless loop within the inner face 90

List of Tables

3.1 Examples of asymmetrical exponential function 29

4.1 Simulation configurations 52
4.2 AODV parameters used in simulation 59
4.3 LEAR parameters used in simulation 63
4.4 GPSR parameters used in simulation 64
4.5 Delivery performance analysis: 300 CBRs, CBR interval = 0.2s/packet, pause time = 0s, full battery initially, simulation time = 1500s 72

Chapter 1

Introduction

A Mobile Ad Hoc Network (MANET) [1] is a collection of wireless mobile nodes dynamically forming a temporary and arbitrary network without the usage of any pre-existing network infrastructure or centralized administration. Possible applications with Ad Hoc networking could be implemented in military battlefield, disaster rescue, sensor networks, rooftop networks, vehicular networks, cell phone call routing, beacon networks (nodes communicate with one another using beacons) and personal-area networks. For example, a sensor network is a network collaborative with many sensor nodes which produces measurable responses to the change in physical conditions. For kinds of wireless sensor networks, they share the same multi-hop routing and energy-optimization issues as MANETs. Therefore, many routing algorithms and energy-saving strategies for MANETs could also be used in wireless sensor networks.

1.1 Motivations

In order to keep communication through the network, mobile nodes in a MANET have to work together to provide some network layer functions, such as packet routing. Due to the mobility of nodes, the network topology is dynamic and the task of finding and maintaining routes becomes difficult. In recent years, many protocols have been proposed with the goal of achieving efficient routing. We can divide them into two groups: topology-

based routing and position-based routing, based on the routing process and the requirement of network topology knowledge. The first group of routing approaches broadcast route requests to obtain the route from the source to the destination. So the source sender knows the whole path to the destination before packets are forwarded. The second group of routing algorithms usually make route decisions only based on the local position information of neighbors and the destination. So, no route request is needed. The whole path from the source to the destination is unknown to the source sender.

Furthermore, mobile nodes are finitely battery-powered. Since Ad Hoc networks are often used in crucial situations where steady network connectivity and a highly efficient routing service are required, it is very important to optimize the energy consumption of mobile nodes so as to prolong the lifetime and maintain the integrity of the whole network. Possible energy consumption optimization strategies include: avoid unnecessary energy consumption when nodes are idle; minimize the end-to-end energy consumption in the packet transmission and balance the energy usage of node to prolong the lifetime of the network. Moreover, routing overhead, such as control messages and routing queries, could also consume a large amount of energy and should be reduced as much as possible.

Based on the above discussion, how to combine power-awareness and efficient routing is a challenging and meaningful topic. In this thesis, we design a new power-aware position-based routing protocol, that uses localized routing and balances energy consumption of nodes. We design this protocol to offer scalable, efficient and low-cost routing for wireless mobile Ad Hoc networks.

1.2 Contributions: Power-Aware Weighted Forwarding (PAWF) Routing Protocol

PAWF, a power-aware weighted forwarding routing protocol proposed in this thesis, is a pure position-based routing approach for MANETs. Since PAWF is a position-based one,

the routing process relies on the knowledge of nodes' geographical positions.

In PAWF, two routing processes are defined: the greedy forwarding process and the recovery process for when the greedy process fails.

Greedy here means that the packet is forwarded to the next hop node with a shorter distance (or positive forwarding achievement) to the packet destination. In the greedy mode, the routing decisions are made hop by hop, only based on the position and battery level of the current node's strict neighbors and the position of the destination. A power-aware weighted forwarding function is used to choose the next hop node, which takes the energy consumption value, the energy residual value and the forwarding achievement into consideration. Therefore, the packet is forwarded to the next hop node with relative high positive forwarding achievement and a good battery level.

When the current node can not find a next hop node with positive forwarding achievement, the greedy process of routing fails. However, this does not mean that the packet forwarding fails and the packet should be dropped. In this case, there could still exist a path from the source to the destination. Therefore, a recovery process is needed to find such a path. In our research, an existing planar graph face-crossing method [2] is used by PAWF to forward packets when the greedy process does not work. The routing of PAWF in the recovery mode also proceeds hop by hop and only requires position information of strict neighbors and the destination of the packet. However, in this mode, the battery status is not considered and paths may not be short.

We evaluated the performance of PAWF using simulation and find that PAWF keeps high routing efficiency, good energy-balancing ability and increases the lifetime of the network greatly. Moreover, PAWF is loop-free in the static network.

In summary, this thesis contributes:

- The design and application of a weighted forwarding function, which combine the energy consumed value, energy residual value with the forwarding achievement.

- The PAWF routing algorithm, which performs forwarding decisions locally and only requires information concerning the current node's strict neighbors and the destination.

- The full PAWF routing protocol, designed for wireless mobility networks, which includes the greedy forwarding process, the recovery process and the neighbor table maintaining method.

- A detailed, simulation-based performance evaluation of PAWF. The performance is explored via simulation using GlomoSim [3]. Experiments are made under different network topologies and energy scenarios and results are compared with three other routing protocols, a topology-based non-power-aware protocol, AODV [4], a position-based non-power-aware protocol, GPSR [2] and a topology-based power-aware protocol, LEAR [5].

- A detailed analysis of the network and energy consumption characteristics where PAWF is expected to work best and worst.

- Further research topics based on the PAWF protocol.

1.3 Layout of the Thesis

The main part of this thesis is the design and implementation of PAWF. We also show that PAWF is able to find global energy-sufficient near-optimal paths by locally optimal choices, avoid the over-usage of nodes and prolong the lifetime of the network.

In Chapter 2, we introduce the structure of the Ad Hoc network, and review existing routing algorithms for MANETs, including regular routing approaches and power-aware routing approaches.

In Chapter 3, the complete PAWF model is described and analyzed. First, we introduce our research goal, and then discuss the assumptions and specifications in our protocol de-

sign. After that, we propose PAWF in detail, including design, components and the whole routing process.

In Chapter 4, we evaluate PAWF by simulations with three different energy scenarios: full battery for all nodes, scarcity of energy and scarcity of energy with node reloading. Experiments are made based on different movement patterns and different traffic loads. We also analyze the different properties of routing performance between topology-based routing, position-based routing and power-aware routing, based on our experimental results.

In Chapter 5, we further discuss the properties of PAWF. First, we describe the characteristics of networks where we expect PAWF to work best and worst. Then, the performance of PAWF and its relation to the energy consumption of nodes is discussed. Finally, we introduce existing location services and their effect on the performance of PAWF.

We conclude in Chapter 6, including reviewing the design of PAWF, summarizing the results of evaluation and presenting the contributions of this thesis. Future directions for work are also presented in this chapter.

Chapter 2

Related Work

A mobile Ad Hoc network is a collection of autonomous nodes or terminals that communicate with each other to a multi-hop wireless network and maintain connectivity in a decentralized way. Since the nodes communicate over wireless links, they have to contend with the effects of radio communication, such as noise, fading, and interference. Every node in a mobile Ad Hoc network functions as both a host and a router, and the control of the network is distributed among nodes. When a node transmits a packet, it simply broadcasts it. Therefore, nodes can overhear other nodes' signals, even if they are not the transmission recipients. The network topology is in general dynamic, because the connectivity among nodes may vary with time due to existing node departures, new node arrivals, and node mobility. In addition, mobile hosts are supplied by a finite battery. So the lifetime of the network and nodes may be limited.

2.1 Ad Hoc Networks

As with wired networks, in general, we can talk about a layered structure of the Ad Hoc network architecture. This thesis mainly contributes to the network layer (the third layer) functionality. Below the network layer are the physical layer and Medium Access Control (MAC) layer. Therefore, a high level discussion of these two layers are given as follows.

As the lowest level, the physical layer provides the services for raw bit transfer over

the medium. Specifications for the physical layer deal with the frequency bands used, modulation techniques and the services provided to the higher layers. The possible physical media for wireless networks include: infrared, microwave and radio. Infrared is simple and inexpensive. They use the same signal frequencies as fiber optic links. Microwave operates at less than 500mW. It uses narrow-band transmission with single frequency modulation and is usually set up to 10 GHz. The usage of radio depends on the bandwidth requirement of service. One common radio signalling method is Direct Sequence Spread Spectrum (DSSS) [6]. DSSS is a modulation method, which maps the data bits to a pattern of chips and then maps them back into bits at the destination. The chipping pattern is redundant for each bit that is transmitted, which increases the signal's resistance to interference.

The medium access control (MAC) layer handles collision detection, fragmentation and acknowledgements. In most wireless protocols, they use Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA or MACA) [7] as the MAC layer protocol. This avoids collisions by checking the channel before using it. If the channel is free, it can start sending. Otherwise, it waits a random amount of time before re-sending. For each retry, an exponential back-off algorithm is used. Moreover, an acknowledgement scheme is also used with the collision avoidance mechanism to inform the successful transmit/retransmit between communication peers.

The protocol we use in our research, 802.11b [6], is a Wireless Local Area Network (WLAN) standard developed by the IEEE's 802.11 working group. At the physical layer, 802.11b uses the DSSS radio transmission method and operates in the 2.4GHz ISM (Industrial, Scientific and Medical) band. At the MAC layer it supports two modes of operations: Distributed Coordination Function (DCF) and Point Coordination Function (PCF). The first function does not use any kind of central control and the second function uses a base station to control all activities. In the mobile Ad Hoc network, there is no base station or other central control equipment. So the DCF mode is used with a CSMA/CA mechanism. Moreover,

the 802.11 standard includes the Request to Send/Clear to Send (RTS/CTS) acknowledge function to control station access to the medium. With RTS/CTS, data transmission is not permitted until the sending station completes a RTS/CTS handshake with the receiving station. The RTS/CTS handshake process is as follows: the sending station initiates the process by sending a RTS frame. If the receiving station receives this RTS, it responds with a CTS frame. The sending station must receive a CTS frame before sending the data frame. The CTS also contains a time value that alerts other stations to hold off from accessing the medium while the sending station transmits its data. The RTS/CTS handshaking provides efficient control for the usage of the shared medium and reduces occurrences of collisions.

2.2 Regular Routing in MANETs

The third layer is the network layer and this is where the thesis makes contributions. In the network layer, a routing protocol is needed whenever a packet is to be delivered over several nodes to arrive at its destination.

Due to the mobility of nodes in MANETS, it is necessary to design efficient routing protocols to allow the nodes to communicate over multi-hop paths. These routing protocols should be robust enough to adapt to dynamic network topologies and avoid using any more of the network (energy or bandwidth) resources than necessary.

Many protocols have been proposed to achieving efficient routing. Based on the routing process and the requirement of network topology knowledge, we can divide them into two groups: topology-based routing and position-based routing.

2.2.1 Topology-based Routing

For topology-based routing algorithms, the route discovery, maintenance and update are based on the knowledge of all or part of the topology of the network. Usually a route query process is used to obtain the whole path from the source to the destination before transmitting packets. So a given amount of routing overhead is always present relative to

the size of the network and the mobility level. When the network is huge and the mobility of nodes is high, the topology of the network will change frequently and lead to high routing overhead.

Among all topology-based approaches, Destination-Sequenced Distance-Vector Routing (DSDV) [8], Clusterhead Gateway Switch Routing (CGSR) [9] and Wireless Routing Protocol (WRP) [10] work like regular wired network routing protocols, building up routing tables or link state tables, maintaining available paths (routes) to all possible destinations, and updating the routes and topology information periodically by broadcasting. Since routes to all possible destinations are maintained and updated with a global broadcast method, the routing overhead of this type of approaches is consequently high.

Another group of topology-based routing algorithms include Ad Hoc On-Demand Distance Vector protocol (AODV) [4], Dynamic Source Routing protocol (DSR) [11] and Temporally-Ordered Routing Algorithm protocol (TORA) [12]. For these approaches, the path to the destination is discovered only when the node has traffic to transmit. So we could call them *on-demand topology-based routing* protocols, or simply *on-demand routing*.

AODV is a typical on-demand routing protocol designed for MANETs. AODV builds routes between nodes only as desired by source nodes. It maintains these routes as long as they are needed by the sources.

An AODV-node informs its neighbors about its own existence by constantly sending "hello" messages at a defined interval. This enables all nodes to know who their neighbors are and keeps the local connectivity of the network.

To resolve a route to some specific node, AODV floods a route request(RREQ) to its neighbor. This RREQ contains the sender address, the destination address, current sequence number, the broadcast ID and the most recent sequence number for the destination of which the source node is aware. The receiving node checks if it has a route to the destination. If a route exists and the sequence-number for this is higher than the supplied

one, a new route is found. The node replies to the request by sending a unicast route reply (RREP). All nodes, that forward this reply back to the source, also create a forward route to the destination. If a route does not exist, the receiving node rebroadcasts the RREQ. Nodes keep track of the RREQ's source address and broadcast ID. If they receive a RREQ which they have already processed, they discard it and do not forward it.

If the source node does not receive a RREP within a time-limit, the destination is thought to be unreachable. If the source node receives multiple RREPs, the newest route could be chosen based on the sequence number. Nodes along the route keep their routing tables updated as long as traffic flows along the route. If not, the nodes will discard the routing entries after a specified time. To be sure that the route still exists, the sender has to keep the route alive by periodically sending packets. All nodes along the route are responsible for the upstream links. So a broken link will be discovered by the closest node. This node signals the broken link by sending an error message upstream so that the source nodes can start to search for a new route.

Based on the process of AODV, as an on-demand approach, it reduces the routing overhead to some degree, because it does not need to keep and update unnecessary routes. However the route discovery process is still implemented by broadcast flooding, so the routing overhead could still be high and is affected by the mobility of nodes.

2.2.2 Position-based Routing

For position-based routing algorithms, the routing decisions are made based on the local position information. Usually, pure position-based routing only requires the position information of strict neighbors and the packet destination. So, no route query process is needed and routing is not relative to the whole topology of the network. At the same time, position-based routing forwards packets hop by hop. So the whole path from the source to the destination is unknown to the routing protocols.

Location-aided or pure position-based approaches are designed usually to reduce the

overhead of routing and increase the scalability of routing protocols. Existing approaches include Location-aided routing (LAR) [13], Distance Routing Effect Algorithm for Mobility (DREAM) [14], Greedy Forwarding rouing (GF) [15], Greedy Perimeter Stateless Routing (GPSR) [2] and GEographic DIstance Routing (GEDIR) [16]. Among them, LAR tries to improve the efficiency of the on-demand routing algorithms by restricting routing discovery flooding within a smaller "request zone". DREAM, GF, GPSR and GEDIR are pure position-based routing approaches, which route the packets hop by hop only based on neighbors' position and the destination position.

GPSR [2], the Greedy Perimeter Stateless Routing, is a robust pure-position-based routing protocol for MANETs. GPSR uses the positions of nodes to make local packet forwarding decisions. For each node, a neighbor table is built up and updated by periodical beaconing. The beacon message is only locally broadcasted and not relayed by the receiving nodes. GPSR uses greedy forwarding to forward packets to nodes that are always progressively closer to the destination. In regions of the network where such a greedy path does not exist, GPSR recovers by forwarding in perimeter mode, in which a packet traverses successively closer faces of a planar subgraph of the full radio network connectivity graph, until reaching a node closer to the destination, where greedy forwarding resumes. This planar subgraph method is also used in our scheme and is described in more detail in Section 3.4.4.

Based on the routing process of GPSR, we can conclude that pure position-based routing could reduce the routing overhead greatly, because it does not maintain routing tables and does not flood route requests globally. Moreover, the routing overhead of position-based strategies is constant and not relative to the frequency of network topology change. Therefore, this type of routing algorithm could potentially work better than topology-based ones in a large-scale high mobility network. Another important property of pure position-based routing is that the routing process proceeds completely locally and the route decision

is made hop by hop. So localized QoS routing, such as power-aware position-based routing, becomes possible.

For pure position-based routing algorithms, accurate position information of destinations is required to make the correct routing decisions. If the destination is not fixed or static, a location service is needed for querying, which could affect the performance of routing and increase the routing overhead. How to design a suitable location service for position-based routing algorithms in MANET is discussed in Section 5.3.

2.3 Power-Aware Routing in MANETs

The Ad Hoc network typically consists of battery-limited nodes, so energy-aware routing becomes important and deserves study. Based on the research in recent years, power-awareness of Ad Hoc routing can be achieved in two aspects: (1) Energy saving and (2) Energy consumption balance of nodes. Approaches combining the above two metrics will be introduced in following sections.

2.3.1 Energy Saving

Based on the existing research work, energy saving in existing routing algorithms could be divided into two types of strategies: minimum-energy routing where the energy for end-to-end transmission is reduced based on the propagation model, or radio-off method where the energy consumption during the idle time is reduce by setting nodes in the energy-saving mode when possible.

Minimum-Energy Routing

In MANETs, energy could be saved by minimum-energy routing, which chooses paths through a multi-hop Ad Hoc network that minimizes the total energy consumed for transmission from the source to the destination. This strategy does not pay attention to the current battery level of the node. If all nodes use the same and constant signal transmission

power (in Watts), the conventional minimum-hop routing will be most energy efficient. However, if we use a more complicated radio, which is capable of adjusting nodes' signal transmission power, things become different. Since the propagation path loss model with the signal transmission power (p) and distance to the next hop node (d) is $p \sim 1/d^n$ where $n \geq 2$, relaying a packet using more intermediate nodes with low signal transmission power may result in lower energy consumption than increasing the signal transmission power to communicate directly.

Bergamo *et al.* [17] propose a Distributed Power Control (DPC) mechanism and a minimum-energy routing protocol, where the link cost is set to the one-hop energy consumption. Classic routing algorithms, such as Dijkstra shortest path algorithm [18] as well as existing Ad Hoc routing schemes, such as AODV or DSR, can be used to compute the path that uses the smallest cumulative energy. In the case where nodes can dynamically adjust their signal transmission power based on the link distance, such a formulation can lead to a path with a large number of hops.

The Power-Aware Route Optimization (PARO) algorithm of J. Gomez *et al.* [19] is also designed for scenarios where nodes' signal transmission power is tunable. PARO attempts to generate a path with a larger number of short-distance hops. According to the PARO protocol, an intermediate node monitors an ongoing direct communication between two nodes and evaluates the potential for energy savings by inserting itself in the forwarding path, and in effect, replacing the direct hop between the two nodes by two shorter hops through itself.

In [20], V. Rodoplu and T. H. Meng present a position-based distributed algorithm aimed at achieving minimum-energy paths from multiple sources to only one destination (they call it the master site node). This algorithm makes the topology control in a distributed fashion and guarantees connectivity of the entire network. Relying on the peers' position information and a simple propagation model with tuning signal transmission power, the

minimum-energy route to the destination is expected.

Research completed by Ivan Stojmenovic and Xu Lin in [21] and by Yuan Xue and Baochun Li in [22] proposes two other power-aware position-based routing algorithms respectively, which depend only on local information of the neighbors and the packet destination. Both of them minimize the end-to-end energy consumption with variable signal transmission power. In [21], energy residual values of nodes are also considered to prolong the lifetime of the network. Since the routing decision is made locally, excessive network communications is not required and corresponding energy is saved. For most proposed power-aware position-based routing algorithms, their research goal is to minimize the end-to-end transmission energy from the source to the destination. This is different from our research goal of this thesis. In our research, we balance the energy usage of nodes and then increase the lifetime of the network, at the same time the routing efficiency is kept as much as possible . With our approach, the end-to-end energy consumption of transmission may not be optimized.

Maximizing Energy-Saving Mode

Another type of approach to reduce energy consumption in MANETs is to set nodes in more energy-saving modes as much as possible. There are four operation modes of nodes: (1) transmitting, where the node sends or transits packets; (2) receiving, where the node receives the packets as the recipient; (3) idling, where the node has no traffic to handle. But since the radio is still on, the node can ***overhear*** the traffic from other nodes even it is not the recipient; and (4) sleeping, where the radio of the node is off and the node stays in the energy-saving mode. So, mobile nodes consume energy not only in transmitting and receiving, but also in idling (by overhearing). In order to reduce unnecessary energy consumption, nodes without packets to send and receive could be switched to the sleeping mode and turn the radio off for energy saving. Radio-off can be controlled based on the information from the MAC layer or upper layers.

I. Batsiolas and Ioanis Nikolaidis [23] present a radio-off scheme based on the information in the transportation layer. They design estimators for the round-trip time and round-trip time variance used by TCP and use them to transfer the transceiver of a mobile node to sleep over extended periods of time when packet activity is not anticipated. Furthermore, the data-link layer information is also used to further control when to set the mobile node's network interface devices to doze and when to wake them up.

Xu *et al.*[24] propose an Adaptive Fidelity Energy-Conserving Algorithm (AFECA), in which the information from above the MAC layer is employed to control the radio-off. This algorithm uses observations about node density to increase the time of radio-off. When many equivalent nodes are able to forward data, they power off for longer intervals.

The Geographical Adaptive Fidelity (GAF) algorithm introduced by Xu *et al.* [25] consist of a conservative energy conservation scheme, in which nodes use geographic location to divide the whole network into fixed square grids. The size of grids keeps constant, regardless of node density and position. In each grid, one node is guaranteed to stays up to route packets. Other nodes in that grid can switch to sleeping and save energy. GAF takes advantage of route redundancy in dense Ad Hoc networks by turning off devices that are not required for global network connectivity. SPAN proposed by B. Chen *et al.* [26] has similar goals to GAF. SPAN also tries to coordinate the sleeping activity of the nodes so that a connecting backbone is always present. The difference of SPAN from GAF shows in two aspects. First, SPAN uses local broadcast messages to figure out changes of the network topology. So no geographic positions need to be known by nodes. Secondly, SPAN integrates with the 802.11 power-saving mode.

The Power-Aware Multi-Access protocol with Signalling (PAMAS) protocol [27] designed by P. Karn is built on the MAC layer. PAMAS uses modified Multiple Access with Collision Avoidance (MACA) [7] protocol and provides a separate channel for RTS/CTS control packets. Energy conservation is achieved by requiring nodes which are not able to

receive and send packets to turn off the radio interfaces. The RTS/CTS messages through the control channel allow for nodes to determine when and how long to power off.

2.3.2 Energy Balance of Nodes

As an alternative (or in addition) to the energy-saving routing, the battery status of intermediate nodes is also a concern, which directly affects the network lifetime and nodes' lifetime. In some recent papers, the residual energy of nodes have been used when considering routing metrics.

Kyungtae Woo *et al.* propose LEAR, a Local Energy-Aware Routing algorithm [5], which achieves a trade-off between balanced energy consumption and shortest routing path. This algorithm is based on DSR. DSR has an on-demand route discovery mechanism, where a route request for a source-destination path is broadcasted. LEAR considers every nodes's willingness to participate in the routing path and to forwarded data packets on behalf of others. Each node accepts the route request messages based on its battery level. So the destination will receive a route request message only when all intermediate nodes along the route have a good battery level.

In order to avoid the over usage of nodes, we could route the message along the path with the maximal minimal fraction of remaining energy. Such a path is called a max-min path. Max-min path routing could balance the energy consumption of nodes, but also increase the end-to-end energy consumption in some situations. Facing this challenge, Q. Li, J. Aslam, and D. Rus [28] present an adaptive max-min path routing algorithm, which combines max-min path routing and minimum-energy routing. The choice of path consists of two steps: (1) Compute a path with minimal energy consumption P ; (2) Compute the path that maximizes the minimal residual energy in the network. Then they try to find a path, which at most uses zP energy for sending a message while maximizing the minimal residual energy fraction. The parameter z is required to be greater than 1 and is adaptively changed based on the energy consumption in the last fixed period time. When $z = 1$, this

approach computes the minimal-energy path. When $z = \infty$, the max-min path routing is achieved.

Minimum-energy routing has also been extended by J.H. Chang and L. Tassiulas [29] to prolong the network lifetime by distributing energy consumption fairly. In their protocol, nodes adjust their signal transmission power and select routes to optimize energy consumption. Suman Banerjee and Archan Misra [30] propose a more complicated extension of minimum-energy routing, which takes link error rate and thus retransmission energy consumption into consideration when attempting to minimize the overall energy of the path. They define an on-demand power-aware routing protocol, MRPC, that integrates a node-specific metric, (residual battery value) and the link-specific metric to select a path with the largest packet capacity at the "critical" node, so as to increase the operational lifetime of the network.

2.4 Chapter Summary

This chapter introduces the architecture of the Ad Hoc network and the functionality of its physical, MAC and network layers. Due to the dynamic topology of the network and finite energy of nodes, it is meaningful to combine Ad Hoc routing with power-awareness. In this chapter, we also present existing power-awareness strategies, such as minimum-energy routing, radio-off and energy balance of nodes.

Chapter 3

Power-Aware Weighted Forwarding (PAWF) Protocol

In this chapter, we present PAWF, a power-aware weighted forwarding routing protocol. Firstly, we introduce our research goal, and discuss the assumptions and specifications in our protocol design. PAWF is designed based on regular position-based routing algorithms plus power-awareness. All components of PAWF and its whole routing process are described in detail in this chapter.

In mobile Ad Hoc networks, there could exist multiple paths between a pair of source and destination. Power-aware routing chooses a path among them which satisfies a predefined energy-aware metric. For position-based routing, it is impossible to know the whole path in advance. We can, however, discover the global near-optimal power-aware path with hop by hop forwarding based on some specific power-aware forwarding function. This is the general idea of PAWF.

3.1 Research Goal

PAWF is a power-aware position-based routing protocol for MANETs, which forwards data packets hop-by-hop only based on the local information of strict neighbors and the destination. It combines the energy consumption situation of nodes and packet forwarding achievement as metrics and tries to balance energy consumption of nodes in the whole

network so as to keep the integrity (connectivity) of the network for a longer time. The inspiration of PAWF comes from two observations: (1) Position-based routing algorithms can significantly reduce the routing overhead and adapt to the mobility and topology changes better than topology-based ones. Adding an energy-aware metric into position-based routing can further optimize the usage of battery and performs well in large-scale high-mobility networks; (2) The energy consumption balancing strategy can efficiently avoid the overusage of specific nodes and deserves more research. Based on the above observations, we propose PAWF to satisfy the following design objectives.

(1) Energy Consumption Balance

Regular position-based routing algorithms do not consider the energy consumption situation as a routing metric. So it could be possible that nodes tend to have widely different energy consumption, resulting in the early death of some heavy-workload nodes; while the batteries of some other nodes maybe still remain intact. With PAWF, mostly, nodes with relatively adequate energy have a higher probability to work for data transmission. So energy consumption of nodes is balanced. Moreover, if we could reduce the energy consumption during the idle time greatly and have the energy consumed during receiving and transmitting dominate the energy consumption of the network, an energy balanced routing protocol could also lead to a workload balance to a corresponding level.

(2) Low Routing Overhead

Position-based routing has low routing overhead for forwarding packets. PAWF completely inherits this advantage. The position and energy consumption information is exchanged only among neighbors by a periodical beacon method and the protocol keeps low and constant forwarding routing overhead. Therefore, only a neighbor table is maintained by each node. No routing table or other global-state tables are needed to be held and updated in PAWF.

(3) Adaptation to High Mobility and Topology Change

PAWF does not have a route discovery process for packet routing. Therefore, the accuracy of neighbors' information will directly affect the routing performance. In PAWF, two extra methods, piggybacking and MAC layer failure feedback, are used to update the neighbor information in time and reflect the real situation of network topology and nodes' status. The details are discussed in Section 3.4.3.

(4) Prolonging The Lifetime of The Network

Since mobile Ad Hoc networks usually work in crucial situations, it is important to keep the integrity of the network for a longer time. PAWF can reach this goal by balancing energy usage of the nodes.

3.2 Assumptions and Specifications

In this section, we discuss the assumptions and specifications of the design of PAWF.

First, we assume all mobile nodes have the same technical parameters, such as transmission range and battery capacity value. We also assume radio connectivity among the mobile nodes is bidirectional, which is required by the IEEE 802.11 [6] MAC layer CTS/RTS acknowledgement.

The protocol requires that the source (or forwarding) node knows the neighbor positions with some accuracy. This is handled by each node using GPS to establish its own position and then to exchange position information with its neighbors. We do not add GPS cost into our routing protocol, because that cost is quite small. Based on [25] and [31], the GPS cost in periodical reporting mode is about equal to the energy consumption of nodes in sleeping mode.

The protocol requires that the packet source also know the destination position. The packet destinations could be fixed and known, or predictable. For example, if mobile nodes

and keep the routing efficiency, rather than reduce end-to-end energy consumption. So a power model with fixed signal transmission power is helpful to avoid additional relays of intermediate nodes and more suitable for our research goal.

3.3 Position-based Routing

In this section, we present the basic idea of combining the power-awareness with regular position-based routing algorithms.

3.3.1 Physical Environment of MANETs

A mobile Ad Hoc network is an autonomous system of mobile nodes connected by wireless links. The nodes are free to move randomly and organize themselves arbitrarily. Thus, the network's wireless topology may change rapidly and unpredictably.

In our research, all the mobile nodes are mapped to a 2-dimensional plane environment. Each node obtains its physical position *(x,y)* from a GPS device. Two nodes can communicate with each other if and only if both of them are located within the radio range of the other. Therefore, two nodes A and B are neighbors and thus joined by a wireless link if and only if the Euclidean distance between their x-y coordinates in the network is less than the minimum of their transmission range (radii). Consequently, the distance between A and B is defined as the Euclidean distance between their x-y coordinates. In our research, all nodes have the same transmission range and so the same radii.

In order to transmit packets from one node to another node, it could be possible that packets from the source sender are relayed by multiple intermediate nodes to finally reach the destination. Therefore, a mobile Ad Hoc network is a multi-hop system where nodes serve as both routers and hosts, and agree to relay each other's packets toward their ultimate destinations. In the real network environment, the packets are transmitted hop by hop using radio signals. The signal encapsulated with the ID of the next hop node is broadcasted by the sender or relaying node and could be overheard by all neighbors. Only the intended

next-hop node will handle that signal, forward it to the next hop node and finally deliver it to the destination. The source sender or relaying node chooses the next hop node based on the existing path (route) or position information of neighbors and the destination. From the current node A and the next hop node B, the signal is considered to traverse the straight line $\overline{AB}$ to reach B. So the signal transmission distance between A and B is also defined as the Euclidean distance between them.

3.3.2 Position-based Routing

Based on our assumptions, every mobile node in our research has the same radio range r. Therefore, an Ad Hoc network can be modelled as an undirected graph $G = (V, E)$, where V is the set of all nodes and E is the set of all edges. $\forall\ n_i,\ n_j \in V$, there is an edge $(n_i,\ n_j) \in E$, if and only if $Euclidean_Distance(i,\ j) \leq r$. In other words, an edge $(n_i,\ n_j) \in E$, if and only if n_i and n_j are neighbors and can communicate with each other directly. Therefore, the edge between nodes n_i and n_j is described as a straight line between n_i and n_j, which stands for the wireless link for signal transmission between two neighboring nodes.

Although we model this as an undirected graph, we still maintain the physical attributes of position in the plane and distance between nodes. A node has information of its position in the plane, its neighbors (nodes within its transmission range) and the distance to them. Since the node are mobile, the graph will change. Nodes update their position information using GPS and the neighbor information by exchanging messages with their neighbors.

For position-based routing algorithms, the routing process proceeds in a completely local and distributed way and only the information of current mobile node, its strict neighbors and the destination is needed. Therefore, for the current node c and its neighbor set N_c, the routing process for c is only relative to the subgraph $G_c = (V_c,\ E_c)$ of G, where $V_c = \{\{c\} \cap\ N_c\}$ and $E_c = \{(c,\ n_i) \mid n_i \in N_c\}$. A position-based routing protocol usually has two modes: a greedy forwarding process and a recovery process (when greedy

forwarding fails). In the following, we define weights on the edges of G_c for the greedy forwarding process.

The greedy forwarding process consists of two parts: (1) Find the candidate set of the next hop node, (2) Choose the next hop node from the candidate set. The next hop node could be the one nearest to the destination [34], nearest to the current node [35], nearest to the straight line from source to destination [36], or based on some other metrics.

In order to formally describe the greedy forwarding routing process, the following two definitions are made based on our research work:

(Definition 1) Forwarding Achievement: Given the current node c and the destination D, the forwarding achievement $Gain(n_i)$ of node n_i is defined as:

$$Gain(n_i) = Distance(c, D) - Distance(n_i, D) \tag{3.1}$$

(Definition 2) Candidate Set of the next hop node: Given the current node c, the neighbor set N_c of c and the destination D, the candidate set N_c' of the next hop node is the set of all neighbors of c, which have shorter distance (positive forwarding achievement) to D than c.

Since only neighbors belonging to the candidate set N_c' are qualified to be chosen as the next hop node in the greedy process, we can further reduce the local graph G_c to be $G_c' = (V_c',\ E_c')$, where $V_c' = \{\{c\} \cap\ N_c'\}$ and $E_c' = \{(c,\ n_i) \mid n_i \in N_c'\}$. We define the weighted forwarding function $f_c(n_i) = Gain(n_i),\ n_i \in N_c'$. The greedy forwarding strategy chooses a candidate with the largest weighted forwarding function value. If we consider the function value $f_c(n_i),\ n_i \in N_c'$ as the weight value of edge $(c,\ n_i) \in E_c'$, the local graph G_c' is finally transformed to be a ***positive edge-weighted undirected local graph.*** Then, the greedy routing process simply compares the weights of edges in G_c' and finds the edge (c, n_i) which has the largest weight value among all edges in E_c'.

In summary, we can define the weighted position-based greedy forwarding routing as: for the local sender c, find the next-hop node n_{next}, such that $f_c(n_{next})$ is the maximum

intermediate nodes, it is necessary to design a strategy to switch the routing process back to the greedy mode when possible.

```
if N'_c ≠ ∅
    enter greedy mode {
        find n_next such that f_c(n_next) is the maximum over all n_i ∈ N'_c
        forward the packet to n_next;
    }
else
    enter recovery mode{
        forward the packet to a "non-greedy" node in N_c of the local graph G_c ;
        go back to greedy mode when possible;
    }
```

Figure 3.2: Pseudocode for position-based routing process

Based on the above discussion, the complete process of the position-based routing is described by the pseudocode in Figure 3.2. Since the routing process is completely localized and the next hop node is chosen only from the local graph G_c or G'_c, it becomes feasible to implement the power-awareness in a local way. So the key part of designing an energy-aware position-based routing algorithm is to define a suitable weighted forwarding function f_c which combines energy consumption issues and the forwarding achievement efficiently. This is the main goal of this thesis.

3.4 Power-Aware Weighted Forwarding (PAWF) Routing

The design of the Power-aware Weighted Forwarding (PAWF) routing protocol completely inherits the property of localized routing defined in Section 3.3. The PAWF model is described in Figure 3.3. Packet forwarding proceeds hop by hop with our power-aware weighted forwarding function when possible. Strict neighbor information (positions and battery status) is exchanged by beacons. This information is updated by periodical beacon exchanges, information piggybacked on data packets and through feedback from MAC layer failures. Destination positions, which could be fixed, predictable or queried by the the

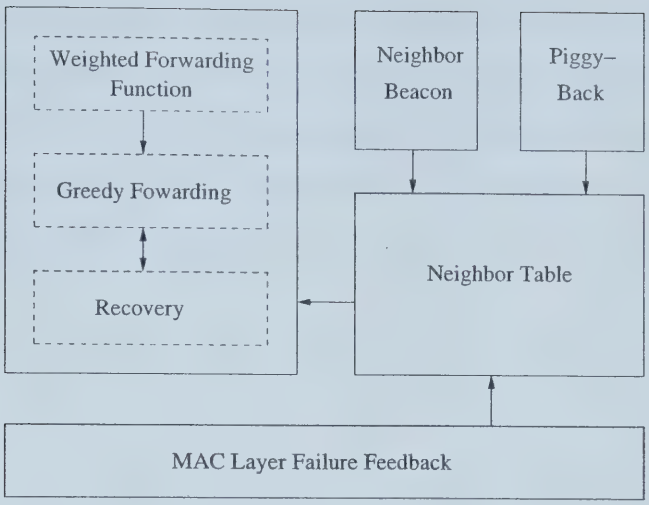


Figure 3.3: Pawf Model

packet sources with some location service, are plugged into the packets and used by all the nodes through the path. Two route discovery modes, greedy forwarding and recovery, are defined with a corresponding mode-switching strategy to solve the local maxima problem.

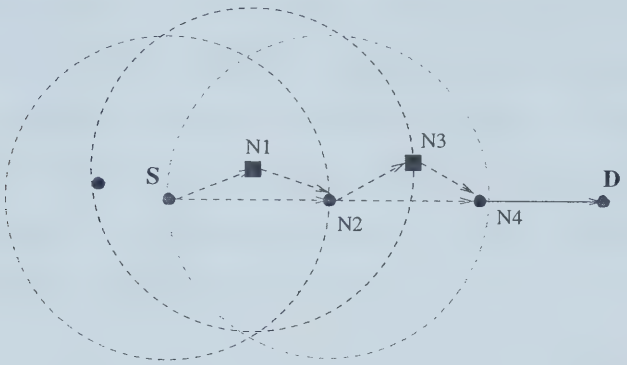
In the following sections of this chapter, we will present each module of the PAWF model, including the principle of the design, the implementation and a relative analysis.

3.4.1 Greedy Forwarding with An Asymmetrical Exponential Weighted Forwarding Function

PAWF mainly works in the greedy mode based on an asymmetrical exponential weighted forwarding function. The next hop node is chosen from the candidate set based on the weight values of forwarding progress and battery capability. As long as the candidate set is non-empty and at least one neighbor is closer to the destination than the current node, the protocol works in the greedy mode.

There are two energy consumption metrics that could be used in our weight function, node residual energy and node consumed energy. The straightforward way to design a weighted forwarding function is to combine the energy consumption metrics and packet

forwarding achievement linearly, which means considering the battery situation and the forwarding achievement in the same level. With this strategy, it could be possible that a neighbor with a small forwarding achievement, but a large value of residual energy, is chosen as the next hop node. Then, the number of hops from the source to the destination could be increased, which would induce higher delay and unnecessary relay operations.



Path1: S–>N2–>N4–>D with the best forwarding achievement
Path2: S–>N1–>N2–>N3–>N4–>D by a linear weighted forwarding function

Figure 3.4: Unnecessary relay with a linear weighted forwarding function

An example of unnecessary relays is shown in Figure 3.4, where the node S is going to transmit packets to the node D. $N1$ and $N2$ are candidates of the node S, and $N3$ and $N4$ are candidates of the node $N2$. We assume that $N1$ and $N3$ (the square nodes) have higher battery level than $N2$ and $N4$ (the circular nodes). If we want the next hop node be the one nearest to the destination, node $N2$ obviously owns the larger forwarding achievement than $N1$ and should be the next hop node of the route. So the final route should be $S \rightarrow N2 \rightarrow N4 \rightarrow D$. But if we choose the next hop node in a linear power-aware way, it could be possible that $N1$ is chosen as the next hop node of S due to its higher energy level and consequently, the final path could be: $S \rightarrow N1 \rightarrow N2 \rightarrow N3 \rightarrow N4 \rightarrow D$. Through the second path, relay operations by A and C are completely unnecessary and should be avoided to reduce the number of hops and delay.

In order to solve this problem and keep our original goal of balancing the energy usage of nodes, we define an asymmetrical exponential weighted forwarding function.

In the greedy forwarding mode, only neighbors belonging to the candidate set are qualified to be chosen as the next hop node. Therefore, for the current node c, we can define the local graph $G'_c = (V'_c,\ E'_c)$, where $V'_c = \{\{c\} \cap\ N'_c\}$ (N'_c: candidate set) and $E'_c = \{(c,\ n_i) \mid n_i \in N'_c\}$. For G'_c and c, let us denote the consumed energy and the residual energy of every candidate n_i as $E_c(n_i)$ and $E_l(n_i)$. The forwarding achievement from c to n_i, $Gain(n_i)$, is defined by *Definition 1* in Section 3.3. The maximum forwarding achievement, the maximum energy consumed and the maximum energy residual value among all nodes in N'_c are expressed as $MaxGain$, $MaxE_c$ and $MaxE_l$ respectively. So our weighted forwarding function is given by:

$$f_c(n_i) = \underbrace{e^{(\frac{Gain(n_i)}{MaxGain})^2}}_{(a)} + \underbrace{e^{(1-\frac{E_c(n_i)}{MaxE_c})}}_{(b)} + \underbrace{e^{(\frac{E_l(n_i)}{MaxE_l})}}_{(c)} \tag{3.2}$$

and the routing process of the current node c is just to find the next hop node n_{next}, such that:

$$f_c(n_{next})\ is\ the\ maximum\ over\ all\ n_i\ \in N'_c \tag{3.3}$$

	X:Forwarding Achievement	Y: Energy Metric	Z: Weight Value
Asymmetry	1	0.5	4.367
	0.5	1	4.000
Near-optimal path with enough Energy	0.9	0.3	3.597
	0.8	0.8	4.122
Avoidance of node over-usage	0.9	0.1	3.353
	0.8	0.6	3.718

Table 3.1: Examples of asymmetrical exponential function

The design of our asymmetrical exponential function is basically derived from the function $f(x, y) = e^{(x^2)} + e^{(y)}$, where we divide $e^{(y)}$ into two parts: (b) and (c) shown in Equation 3.2. (b) and (c) stand for energy consumption and energy residual respectively and are coincident among nodes. Coincidence here means: for node n_i and n_j, if (b) of n_i is

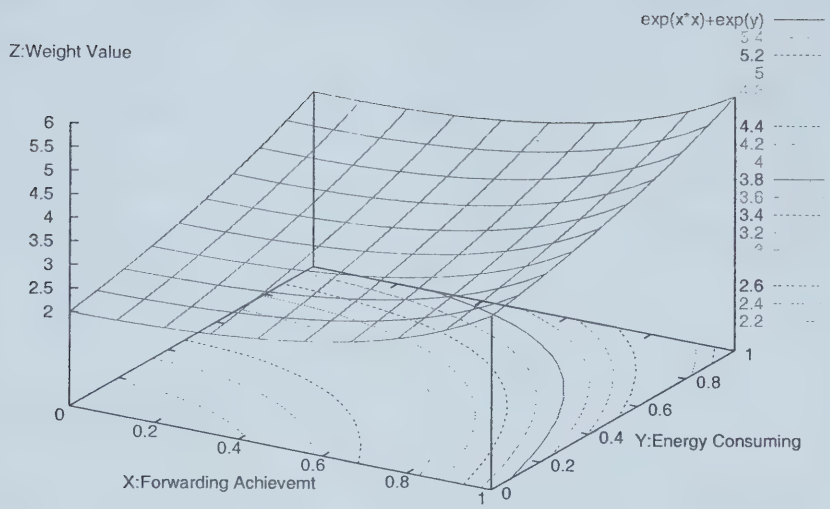


Figure 3.5: Asymmetrical exponential weighted forwarding function

greater than (b) of n_j, (c) of n_i is also greater than (c) of n_j, and vice versa. This property is important to avoid the counteracting effect between the two power-aware parts of our function. The function $f(x, y)$ is plotted by a 3D graph, shown in Figure 3.5, to describe the basic characteristics of our weighted forwarding function. The X, Y and Z axes present the ratio of the forwarding achievement over the maximum forwarding achievement, the energy consumption ratio (part b or c of our forwarding function) and the accumulated weight value respectively.

From the contours on the base of Figure 3.5, we can find that the weight values of Z are not symmetrical between X and Y. Based on the first example in Table 3.1, when the forwarding achievement ratio X is the highest (1) and the energy consuming level Y is median (0.5), the weight value Z is very high, nearly equal to 4.4. But when the X value is median (0.5), even if the Y value is equal 1, the Z value is only 4.0. Therefore, the candidate with the largest forwarding achievement has a very high probability to be the

winner of the next hop node competition. With such an asymmetric property, the number of hops from the source to the destination is usually kept near-optimal and unnecessary relays are reduced greatly.

```
RoutingPAWFGetNextHopGreedy(currentNode){

  thisDistance = GetDistance(currentNode, Destination);

  for all neighbors in the neighbor table of currentNode{
      find maxEnergyLeft;
      find maxEnergyConsumed;
      find maxGFDistance;
  }

  if (maxGFDistance <= 0)
     return NULL;

  maxWeight = 0;
  for each neighbor N_i in the neighbor table of currentNode
        {
      if (neighbor node N_i has positive forwarding
            achievement){

          calculate weightForwarding;
          calculate weightEnergyConsumed;
          calculate weightEnergyLeft;

          weightEnergy = weightEnergyConsumed +
                weightEnergyLeft;
          weightTotal = weightForwarding + weightEnergy;

          if(weightTotal > maxWeight){
              maxWeight = weightTotal;
              nextHop = neighbor node N_i ;
          }
      }
  }

  return nextHop;
}
```

Figure 3.6: Pseudocode for greedy power-aware weighted forwarding

Besides characterizing the asymmetry, Table 3.1 also shows examples of the power-awareness of our function. For example, a node with forwarding value 0.8 and energy value 0.8 has a higher weight value than one with higher forwarding of 0.9 and lower energy of

```
Beacon message:
    pktType,
    myAddr,
    myPosition,
    energyLeft,
```

Figure 3.7: Structure of a beacon message

```
Neighbor entry:
    myAddr,
    myPosition,
    lastUpdatedTime,
    energyLeft,
```

Figure 3.8: Structure of a neighbor table entry

0.3. Therefore, our function usually chooses a next hop node as the one with optimal/near-optimal forwarding achievement and enough energy residual and finally ensures the whole path is near optimal and energy-sufficient. Since nodes with insufficient energy are kept from the routing work, the energy consumption is balanced. Consequently, from the second and third examples in Table 3.1, the over-usage of nodes is usually avoided.

The implementation of the greedy forwarding of PAWF is shown in Figure 3.6. In order to use the weighted forwarding function, the current node first find the values of the maximum greedy forwarding achievement (*maxGFDistance*), the maximum energy left (*maxEnergyLeft*) and the maximum Energy consumed (*maxEnergyConsumed*) among all its neighbors. If the value of *maxGFDistance* is equal to or less than 0, it means the candidate set of the current node is empty and a $NULL$ value is returned to indicate the failure of the greedy forwarding. If the candidate set is not empty, the weight value is calculated for every candidate based on the Function 3.2. Finally, the candidate with the largest weight value is chosen as the intended next hop node and its address is returned. If more than one candidate has the same largest weight value, the next hop node is chosen randomly among them to break the tie.

3.4.2 Beacon Method to Exchange Neighbor Information

In position-based routing, it is necessary to have accurate neighbor information. Periodical beaconing is the common method for position-based routing algorithms to exchange neighbors' information. Beacon messages are only broadcasted to strict neighbors of a node and not relayed further by a receiver. Besides position information, our protocol also requires energy information of nodes to be exchanged by beacon messages.

```
RoutingPAWFReceiveNbrBroadcast(currentNode , nbrBdcst){

  delete all neighbor entries not updated within 2*
        NBR_BDCST_INTERVAL;

  If the sender of nbrBdcst exists in the neighbor table of
  currentNode{
      update this neighbor entry{
          thisEntry.myPosition = nbrBdcst.myPosition;
          thisEntry.energyLeft = nbrBdcst.energyLeft;
          thisEntry.lastUpdatedTime = currentTime;
      }
  }
  else{
      generate a newEntry{
          newEntry.myAddr = nbrBdcst.myAddr;
          newEntry.myPosition = nbrBdcst.myPosition;
          newEntry.energyLeft = nbrBdcast.energyLeft;
          newEntry.lastUpdatedTime = currentTime;
      }
      insert newEntry into the neighbor table;
  }
}
```

Figure 3.9: Pseudocode for beacon receiving and neighbor table update

The structures of a beacon message being broadcasted and neighbor table entry used in PAWF are depicted in Figure 3.7 and Figure 3.8. The beacon message includes the position information and energy residual value of the sender. Since every node has the same technical parameters, the energy consumed value is equal to $BATTERY_CAPACITY - energyLeft$. The beaconing process consists of two parts: beacon broadcasting and beacon receiving. In beacon broadcasting, the node generates a beacon message con-

taining the information defined in 3.7, and then broadcasts to its strict neighbors. In order to avoid synchronization of beaconing, the next beacon broadcast time is set as $beaconInterval + (a\ random\ value\ within\ [0, beaconInterval])$. Beacon receiving process is shown in Figure 3.9. Every node builds up a neighbor table containing all its strict neighbors. One neighbor stands for a neighbor entry in the neighbor table. Nodes keep and update neighbor entries of their neighbor table based on beacons received. A new neighbor entry is added if no corresponding neighbor to the beacon message is found in the table. The neighbor entries not updated within two beacon intervals will be deleted from the neighbor table.

3.4.3 Adaptation to High Mobility and Topology Change

The accuracy of neighbor tables is key to the correct working of position-based routing algorithms. In a mobile Ad Hoc network, the topology of the network changes frequently. Simple periodical beacons sometimes can not ensure in-time updating of the neighbor table and can not reflect the real network topology. For example, some neighbor entries could become stale due to the movement of corresponding nodes. Such a problem becomes more serious in high mobility networks or when some nodes run out of energy and suddenly turn off without notice. Therefore, we implement two extra neighbor table update strategies in PAWF: MAC layer failure feedback and piggybacking. Both of them are low cost and highly efficient.

In our simulation, the IEEE 802.11b standard with DCF is followed. The MAC layer uses CSMA/CA control with a CTS/RTS scheme. If no CTS is received after RTS, and no ACK is received after exceeding a maximum retry count of retransmissions of a unicasted packet, MAC layer will inform the network layer of the packet delivery failure with a control message containing the intended next hop node address. PAWF is then able to delete the corresponding stale neighbor entry.

Such a method is based on an existing MAC layer acknowledgment strategy and will

not expend extra bandwidth and memory. A similar method was used in DSR [11] to update routing tables and in GPSR [2] to maintain neighbor tables.

Piggybacking

The piggybacking method attaches the local sender's position and energy consumption information into all the data packets. In our network, we assume that all the nodes work in non-promiscuous mode. So only the packet's intended next hop node will grasp the piggybacked information and use it to update the neighbor table.

The process of handling piggybacked information is very similar to the beacon receiving process. When the intended next hop node gets a packet, it will check its neighbor table if the packet sender exists in the table. If it exists, the corresponding neighbor entry will be updated using the position and energy information carried by the packet. If there is no neighbor entry, a new neighbor entry will be created and added to the neighbor table. The piggybacking method is a good supplement to the regular neighbor table update, although it increases the packet size.

3.4.4 Recovery of Greedy Forwarding Failure

When the current node does not have a neighbor with positive forwarding achievement, the greedy forwarding process falls into the local maxima and greedy routing fails, even if a path does exist from the source to the destination. In order to overcome this problem, PAWF uses the same recovery strategy as proposed in GPSR, called the planar graph face-crossing method [2]. With this method, the routing decision is still made locally based on the knowledge of the neighbors and destination's positions. The packets are forwarded progressively by a right-hand rule on the faces of a planar graph, which is generated in a distributed way among the current node and its neighbors. The path found in the recovery mode may not be the shortest path, nor does the process consider energy. Therefore, if the packet reaches a node which is closer to the destination than the node at the entrance of

recovery process, the packet forwarding switches back to the greedy mode.

Right-Hand Rule

The right-hand rule is a well-known graph traversing method. The general idea is: *when arriving at a node x from a node y, the next edge traversed is the next one sequentially counterclockwise with respect to x from the edge (x, y)* [2].

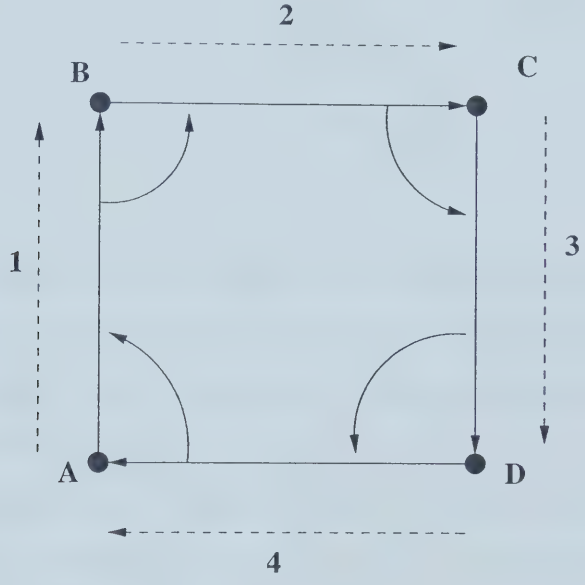


Figure 3.10: The right-hand rule traverses the interior of the square. B receives data from A first and the traversing path is: $A \rightarrow B \rightarrow C \rightarrow D \rightarrow A$

An example is shown in Figure 3.10, where B receives a packet from A, and forwards it to its first neighbor counterclockwise from itself, C. Consequently, the traversing path through the square with a right-hand rule is: $A \rightarrow B \rightarrow C \rightarrow D \rightarrow A$. The right-hand rule is a method to decide the next hop node regardless of the forwarding achievement and will be used in the following face-crossing algorithm.

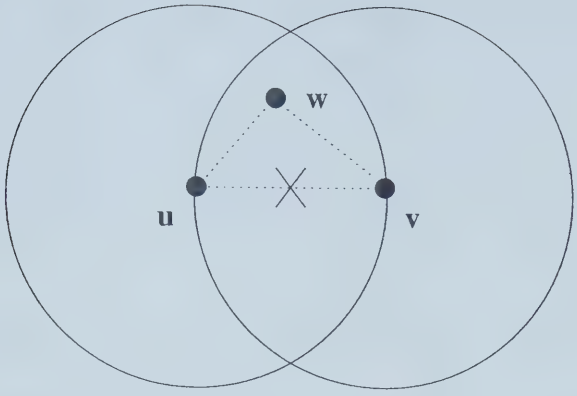


Figure 3.11: The RNG graph

Planar Graph

If a graph G can be given a pictorial representation by means of lines and points in such a way that the lines only intersect at vertex points, then G is call a planar graph [38]. Within the set of planar graphs, a special subdivision is defined as planar straight-line graphs, which stands for the set of graphs that can be embedded in the plane without crossings in which every edge in the graph is a straight line segment [39]. Therefore, in a planar straight line graph, an edge is a straight line connecting two vertices and non-crossing edges mean that any pairs of straight lines have no common points except vertex points. For example, in Figure 3.12, graph a, b and c are all planar graphs. But only graph c is a planar straight line graph.

In our research, all mobile nodes are mapped to a 2-dimensional plane environment. Therefore, the accurate position of the node in the plane is known (at least for the moment). Nodes communicate with one another through a straight wireless link. Therefore, in the recovery process of PAWF, the original graph of the Ad Hoc network is reduced to a planar straight line graph in a distributed way. For simplification, we just call it a planar graph.

The Relative Neighborhood Graph (RNG) [40] is a well-known planar straight line graph. A RNG can be generated from the original graph by deleting edges that cross in

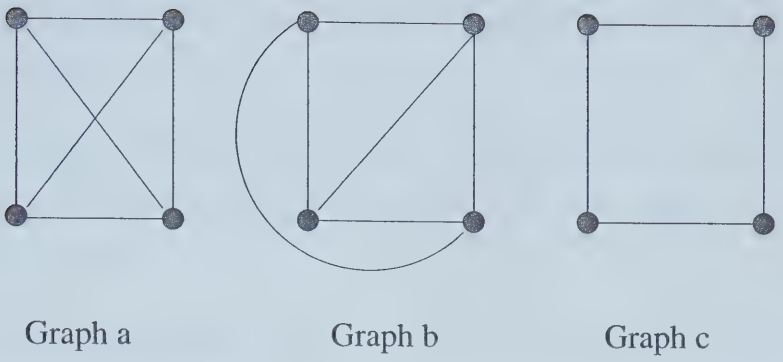


Figure 3.12: planar graph and planar straight line graph

the physical plane. Based on Figure 3.11 (from the paper [2]), a RNG graph is defined as: for two neighbor nodes, u and v, we can draw two circles of radius $r = Distance(u, v)$ and centered at u and v. The edge (u, v) exists in the RNG graph, if and only if no node w appears in the overlapped region of two circles. In a formal way, for existing edge (u, v), it appears in the RNG graph, if and only if $\forall\ w(w \neq u, v),\ Distance(u, v) \leq MAX(Distance(u, w),\ Distance(v, w))$

One important property of the planar graph is: removing edges from the original graph to reduce it to the planar graph will not disconnect the graph. For the RNG, the simple proof is given as: from Figure 3.11, the edge (u, v) is eliminated from the graph only when there exists a w within the radio range of both u and v. Thus, deleting an edge (u, v) requires the existence of an alternate path $u - w - v$ through a node w. So connected components in the original graph will not be disconnected by removing edges not in the RNG.

In the RNG, we name polygonal regions in the graph as *faces*. The face can be closed or not closed. A simple RNG graph is shown in Figure 3.13, where the face 1 and 3 are closed, and the face 2 is not closed.

In summary, the usage of RNG graph in the recovery process are based on following reasons:

- The generation of RNG can be implemented in a completely distributed way. Only

the position information of strict neighbors is needed. So it is applicable for position-based routing.

- The generation of RNG does not destroy the connectivity of the original graph. That is why a RNG is available to be used to handle the local maxima problem in the recovery process.

- No crossing edges exist in a RNG, which ensures the successful traversal of the faces by the right-hand rule.

- Crossing faces aggressively from the source to the destination guarantees the discovery of a feasible path from the source to the destination and successfully recovers from the local maxima problem of greedy forwarding. The face-crossing method is introduced in the next section.

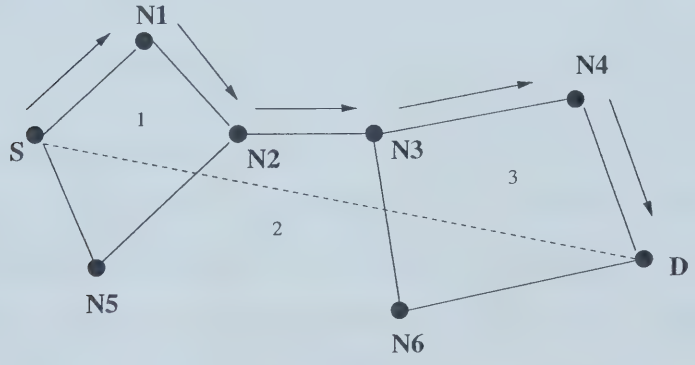


Figure 3.13: Face-crossing traversal: $\overline{SD}$ crosses the face 1, 2 and 3 progressively from S to D, the arrows show the path chosen by the rule of face-crossing

The Reovery Process: Face-Crossing Algorithm [2]

With the definition in Section 3.3, an Ad Hoc network can be described as a graph $G = (V, E)$. We know, if u and v can communicate with each other, they must both know all nodes within the overlapped area. Therefore the generation of the portion of RNG from the

```
PlanarizePAWF(u){
   for each neighbor w of node u{
      for each neighbor v of node u{
         if ( w == v)
            continue;
         else{
            if(Distance(u,v) > Max(Distance(u,w),
                                   Distance(v,w))){
                  delete edge(u,v);
                  break;
            }
         }

      }
   }
}
```

Figure 3.14: Pseudocode for the generation of RNG

graph of the Ad Hoc network can be done in a completely local way. For the current node (denoted by u) and its neighbor table, the non-RNG edges can be removed based on Figure 3.14, where the edge (u, v) is deleted if:

$$\exists\, w \neq u, v : distance(u, v) > Max[distance(w, u),\ distance(w, v)] \qquad (3.4)$$

Since the RNG keeps the connectivity of the graph G, a face-crossing algorithm is design to obtain the feasible path from the source to the destination in the RNG, instead of in G. Such an algorithm guarantees to discover the feasible path and only the local neighbor information is needed. Therefore, this algorithm is suitable to be used in the position-based routing, when greedy forwarding is not available.

An example of face-crossing is shown in Figure 3.13, where the packet is transmitted from S to D. The general idea is: the packet is forwarded on progressively closer faces, each of which is crossed by the line $\overline{SD}$. In detail, the face-crossing algorithm is described as:

- First, at the first node S of the recovery process, the packet is forwarded to the first edge counterclockwise about S from the line $\overline{SD}$, which is $(S, N1)$. This determines

the first face through which to forward the packet, and then the position of node S is recorded as the entrance node of the recovery process and the first intersection point of the faces by $\overline{SD}$.

- Thereafter, the packet is forwarded along the interior of face 1 by the right-hand rule: forward the packet through the next edge counterclockwise from the edge on which it arrived. Whenever $\overline{SD}$ intersects the edge along which a packet is about to be forwarded, check if this intersection is closer to the destination than the intersection previously encountered. We call such an intersection the progressive intersection. If it is true, we switch to the new face (nearer to the destination) bordering on the edge which the packet was about the traverse. The packet is then forwarded through the next edge counterclockwise to the edge it was about to be forwarded through before switching faces. In this example, when traversing face 1, the line $\overline{SD}$ intersects with the edge $(N2, N5)$, where the intersection position is nearer to the destination than the last intersection position (S). So a face change is made and we traverse face 2 from the node $N2$. Based on the same rule, the final path passing through is: $S \rightarrow N1 \rightarrow N2 \rightarrow N3 \rightarrow N4 \rightarrow D$.

3.4.5 The Full Routing Process of PAWF

Now we present the full routing process of PAWF, which mainly proceeds with power-aware weighted forwarding when possible and uses the planar graph face-crossing process when greedy forwarding fails.

Recall that all nodes maintain a neighbor table, which stores the addresses, positions and energy consumption information of their strict neighbors. This table provides all information needed for routing decisions of PAWF in the greedy forwarding mode, except the destination position which is known or queried by the original source sender.

The information required for PAWF's recovery process is carried by the data packets.

```
addrD :              Destination address
positionD :          Destination position
positionE :          Node position entering the recovery mode
positionF :          Position on line(positionE , Destination)
                        when packet entering current face
addrLastHop :        Previous hop node address
positionLastHop :    Previous hop node position
hop0 :               First hop traversing on current face
mode :               Forwarding mode (GREEDY/RECOVERY)
```

Figure 3.15: Packet header of PAWF for the recovery process

A portion of the PAWF header for the recovery process is shown in Figure 3.15, which is the same as that used in GPSR. The destination position is queried by the source sender and used by all the following nodes without change. The $positionE$ is the node position where the packet forwarding enters the recovery mode. The intersection position $positionF$ need not be located at a node. It records the position on $\overline{ED}$ shared by the previous and current faces. $hop0$ is the first hop traversing on the current face, which is used to verify destination-unreachability in the recovery process.

Compared with the greedy forwarding process, the recovery process needs more position information, such as the position of the recovery-mode entrance node and the intersection point of the previous face. It also needs more CPU resources to handle the generation of the local planar graph and face-crossing. At the same time, the recovery process is not power-aware and may choose a longer path to the destination. Therefore, a switching strategy is needed to put the packet forwarding back to the greedy mode when possible. In our research, if the packet reaches a node which is closer to the destination than the entrance node of the recovery process, the packet forwarding switches back to the greedy mode. Since the starting node of the new greedy process is nearer to the destination than the final node of the last greedy process, the local maxima problem is solved.

Based on the above discussion, the full routing process of PAWF is given by Figure 3.16. For the source sender of the packet, it sets the forwarding mode of the packet as $GREEDY$ initially. When a node receives a packet, it first checks if the packet enters one

```
RoutingPAWF(currntNode , msg){
  if (currentNode == destNode)
      msg received;
  if (destNode in the neighbor of currentNode)
      forward msg to destNode;

  switch(msgHeader.mode)
    case GREEDY:
      nextHop = RoutingPAWFGetNextHopGreedy(currentNode);
      if ( nextHop != Null)
          forward msg to nextHop;
      else{
          PlanarizePAWF(currentNode);
          msg.mode = RECOVERY;
          update the header of msg{
              msg.positionE = currentNode.position;
              msg.positionF = currentNode.position;
              msg.addrLastHop = currentNode.address;
              msg.positionLastHop = currentNode.address;
              nextHop = RecoveryInitForward(currentNode ,
                          msg, msg.positionD);
              msg.hop0 = (currentNode ,nextHop);
          }
          forward msg to nexthop;
      }
    case RECOVERY:
      if (GetDistance(currentNode , msg.positionD)
            < GetDistance(msg.positoinE , msg.positionD)){
          msg.mode = GREEDY;
          greedy forward msg;
      }
      else{
          PlanarizePAWF(currentNode);
          nextHop = RecoveryRightHand(currentNode , msg,
                lastHop);
          if(msg.hop0 == (currentNode , nextHop))
              drop msg, destination is unreachable
          else{
              if ((i = Intersect((currentNode , nextHop),
                   (msg.positionE , msg.positionD)))!= NULL)
                  if(Distance(i ,msg.positionD) <
                     Distance(msg.positionF , msg.positoinD))
                       nextHop = FaceChange(currentNode ,
                              nextHop);
          }
          update the header of msg
          forward msg to nextHop;
      }
}
```

Figure 3.16: Pseudocode for the full routing process of PAWF

of the following "easy situations":

- If the current node is the packet destination, the packet transmission is successful.
- If the destination is one of the current node's neighbors, the packet is transmitted to the destination directly.

If the packet does not find the above situations, the PAWF routing process has to be taken.

- If the packet is in the $GREEDY$ mode, the function *RoutingPAWFGetNextHop-Greedy()* is called to forward the packet greedily based on our weighted forwarding function. This chooses an energy-sufficient neighbor that is closer to the destination. If the greedy forwarding does not work (when no neighbor is closer to the destination), the mode of the forwarding is set to be $RECOVERY$, the header of packet is updated and the function $RecoveryInitForward()$ is called to forward the packet based on face-crossing.
- If the packet is in the $RECOVERY$ mode, the current node first checks if the forwarding could go back to the $GREEDY$ mode. If yes, greedy forwarding is chosen again. If the answer is no, the function $RecoveryRightHand()$ is called to choose the next hop node by traversing the current face based on the right-hand rule. After that, we will check if the next hop of the packet is equal to the first hop ($hop0$) of the current face. If it is true, it means a loop happens in the face traversing and the destination is physically unreachable. We should note that, in the loop check, a hop is an operation traversing from one vertex to another vertex of the edge. So, for the edge (a, b), the hop from a to b is not the same as the hop from b to a. If a loop does not happen, the function $FaceChange()$ is then used to check if a progressive intersection happens and a face change is needed. Finally the packet header is updated and the packet is forwarded to the next hop node in the $RECOVERY$ mode.

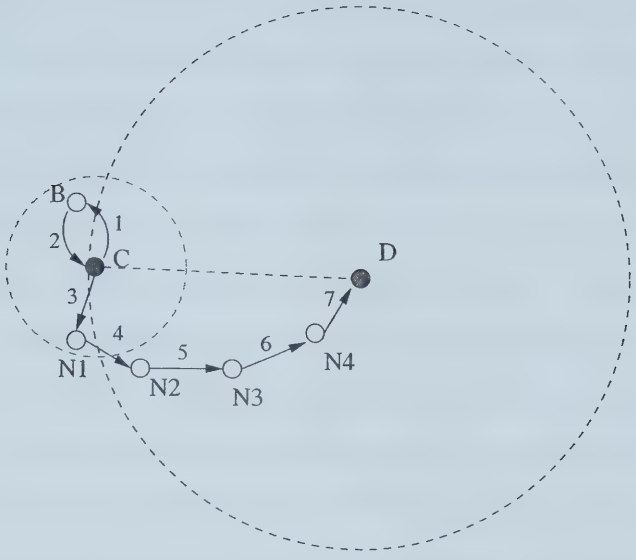


Figure 3.17: PAWF routing process

Let us show an example of how PAWF works to forward a packet to the destination even if a local maxima happens. If a packet arrives at node C, in Figure 3.17, the routing falls into the local maxima. Based on the right-hand rule, node B is the next hop node counterclockwise about C from the line $\overline{CD}$ and the first hop through the current face is $\overrightarrow{(C, B)}$. Since B is not nearer to D than C, the recovery process continues and the packet will be forwarded back to C. The next hop $\overrightarrow{(B, C)}$ is not equal to the first hop of the current face, $\overrightarrow{(C, B)}$, so this is not a loop and the packet reaches node C. Tracing with the right-hand rule and the face-crossing algorithm continuously, the packet is forwarded to node $N1$ and then $N2$. Since $N2$ has a shorter distance to D than the recovery entrance node C, the packet forwarding switches back to the greedy mode and transmits the packet greedily to $N3$, $N4$ and finally the destination D.

It is easy to prove the PAWF routing is loop-free in the static network environment. In the greedy mode, with PAWF, packets are routed greedily and always move closer to the destination. Since there is no mobility, no nodes could be visited more one once during a

packet transmission and the path traversed by PAWF is definitely loop-free. In the recovery mode, we know that the exit point must have a shorter distance to the destination than the entrance point. So if we just consider the recovery process as a pair of nodes: the entrance node and the exit node, it still can be considered as a greedy forwarding operation and can not generate a loop. So what remains is to show that PAWF can reach an exit node without a forever loop happening in the recovery process. In recovery mode, the packet traverses a series of adjacent faces, each by the right-hand rule. Because the graph is planar, traversal of a face by the right-hand rule must visit every edge of the face. If the destination is reachable, there must exist a sequence of adjacent faces along the line between the entrance node and the destination. With the right-hand rule and face-crossing method, these adjacent faces will be traversed in sequence, until reaching the destination. If the destination is disconnected, then PAWF drops the packet when it makes a hop which is the same as the first hop ($hop0$) on the current face. So there is no forever loop in the recovery routing process. Thus we can conclude that PAWF is loop-free in static networks. Above proof is based on the discussion of face-crossing routing in [2].

When nodes are mobile, it could be possible that some neighbors move out of the radio range of the current node before the neighbor table is updated. So the corresponding neighbor entries become stale and could induce some delivery failures or temporary loops. PAWF has used some additional neighbor table update strategies, such as piggyback or MAC layer failure feedback, to keep the neighbor tables as up to date as possible. But it can not guarantee this. Therefore, even thought our experimental results show PAWF works well in the mobility environment, it does not guarantee completely loop-free routing when nodes are mobile. Detailed discussions about PAWF performance with node mobility can be found in Section 5.1.

In order to combine PAWF with the OSI ISO reference model [41], based on Figure 3.18, Simple APIs are used between different neighboring layers, especially between the

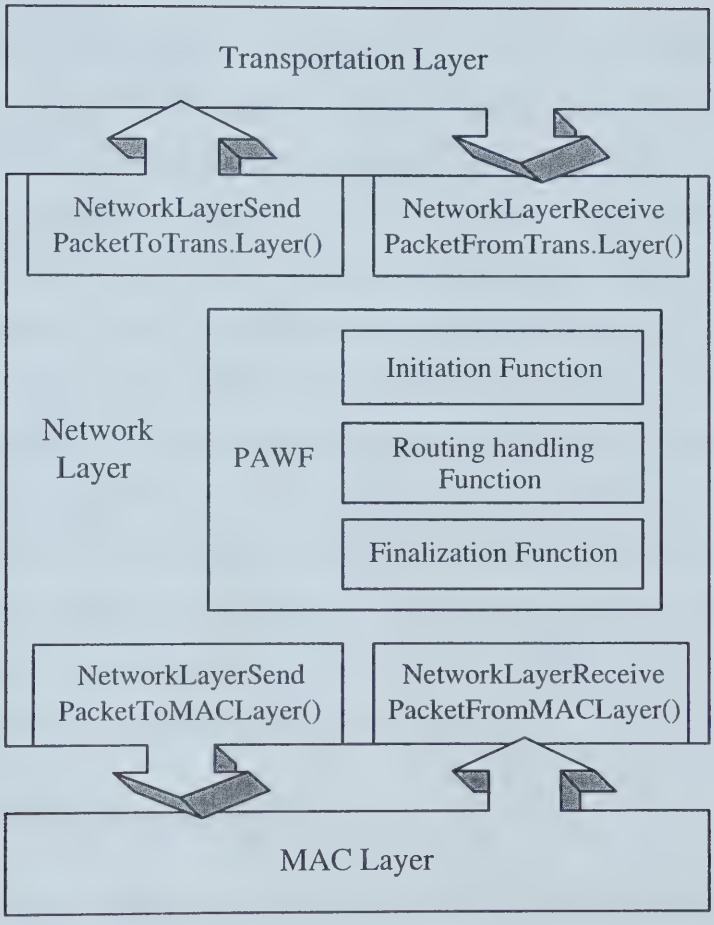


Figure 3.18: The multi-layer design of PAWF and the APIs

MAC and network layers and between the network and transportation layers. These APIs specify parameter exchanges and services between adjacent layers, including transmitting, receiving data or control messages from the upper or lower layer. With the usage of API functions and the multi-layer design strategy, PAWF is easy to be integrated into existing industry applications.

In summary, mobile nodes with PAWF protocol work as shown in Figure 3.19. First, the neighbor table is updated periodically to offer fresh position and battery status of strict neighbors. The triggered neighbor table updates are also available with MAC layer failure feedback or piggybacked information. If a new data packet is generated by some node and needs to be delivered to the destination, the next hop node is chosen based on our weighted forwarding function, with the position and battery status of strict neighbors and the position of the destination. Then, the battery status of the source sender and the position of the source sender and the destination are carried by the data packet and the packet is transmitted to the next hop node. when a data packet is received by some node, if that node is the packet destination, the delivery is successfully done. If the destination is a neighbor of that node, the packet is forwarded to the destination directly. Otherwise, that node piggybacks its own information into the data packet and relays it to the next hop node based on the transmission mode (GREEDY or RECOVERY).

3.5 Chapter Summary

This chapter presents PAWF, a power-aware position-based routing protocol for MANETs, to balance the energy usage of nodes, increase the life time of the network, ensure the efficiency of routing and keep low routing overhead. PAWF routing only requires the information of neighbors and the destination. It achieves the power-awareness with a weighted forwarding function and gets over the local maxima by local planar graph face-crossing. PAWF guarantees loop-free routing in the static network and works well with node mobil-

```
PAWFProtocol(currentNode){

  if ( current time = next beacon broadcast time){
      broadcast neighbor beacon message;
      set the timer for next broadcast;
  }

  if (receive a beacon message){
      if (corresponding neighbor entry exists)
         update that neighbor entry;
      else
         generate a new neighbor entry;
  }

  if (receive MAC layer failure feedback)
      delete the stale neighbor entry;

  if (a new packet is generated){
      piggyback position of destination;
      piggyback position and battery status of own;
      send packet to the next hop node in GREEDY mode;
  }

  if (receive a packet from some other node){
     update neighbor table with piggybacked information;
     if (the packet destination){
        send packet to the upper layer;
        packet delivery is successful;
     }
     else if {the destination is one of neighbors){
              piggyback position and battery status of own;
              forward packet to the destination directly;
          }
     else{
         piggyback position and battery status of own;
         send packet to the next hop node based on Tx mode{
             transmission in GREEDY mode;
                  or
             transmission in RECOVERY mode;
         }
     }
  }

}
```

Figure 3.19: Nodes with PAWF protocol

ity. With the multi-layer design and the usage of APIs, PAWF is easy to integrate and adapt to the standard OSI ISO reference model.

Chapter 4

Simulations and Experiments

In this chapter, simulations are used to evaluate the performance of our protocol, PAWF. We first introduce the simulation environment and the principles of the experiment design. After that, the performance of PAWF is compared with one greedy non-power-aware algorithm, GPSR, one topology-based routing algorithm, AODV, and one power-aware topology-based routing algorithm, LEAR, under different energy scenarios and network topologies.

4.1 Simulation Environment

We implemented PAWF on GloMoSim2.03 [3], a scalable discrete-event simulator for wireless and wired networks. GloMoSim was developed by UCLA and supports a wide range of Ad Hoc routing protocols as well as realistic propagation, radio and MAC layer protocols. The general simulation configuration is shown in Table 4.1.

4.1.1 Physical Layer and MAC Layer Settings

In the physical layer, a TWO-RAY model is used for signal propagation. This model considers both the short direct path and long ground reflection path. It is shown in [42] that this model gives more accurate prediction at a long distance than the simple free-space model. Two-Ray model uses free space path loss (2.0) for near sight and ground reflect earth path

Routing Protocols	PAWF, GPSR, AODV, LEAR
Application Layer	CBR
Network Layer	IP
MAC Layer	802.11 DCF
Radio Layer	RADIO_ACCNOISE
Propagation Layer	TWO-ARRAY
Traffic Mode	Random CBR streams
Packet Size of CBR Streams	512 byte
Wireless Card	Lucent 802.11WaveLan
Bandwidth	2 Mbps
Radio Frequency	2.4e9
Radio Range	250m
Energy Consumption	Tx (5V, 285mA), Rx(5V, 185mA)
Battery Capacity	5V, 2000mAh/normal distributed
Node Number	100
Node Distribution	Uniform
Mobility	Random waypoint
Speed	0 – 10 m/s
Simulation Area	(1200m, 800m)
Simulation Time	1500/3000 seconds

Table 4.1: Simulation configurations

loss (4.0) for far sight. Therefore, for the short direct path, the two-ray model receives signal whose power decays with d^{-2}(d : distance to the destination), and for the long ground reflection path, the signal power decay increases to d^{-4}. The antenna height is hard-coded in the model (1.5m). The implementation of TWO-RAY propagation model in GlomoSim is based on [42].

The radio model is set as RADIO-ACCNOISE, which refers to the standard radio model. This models loss of signal due to noise. Radio with accumulated noise calculates the noise level as the sum of the thermal noise and all signals not successfully received on the same channel. This works with a Signal to Noise Ratio (SNR) bounded packet reception model, which determines successful packet receptions by comparing the yielded SNR with the threshold set in the configuration. This comparison is performed every time the noise level changes. Once the signal being received is determined to be failed, it will not be forwarded to the upper layers.

The MAC layer is modelled based on IEEE 802.11b, using DCF mode with CSMA/CA control and RTS/CTS scheme. This protocol is described in Section 2.1.

4.1.2 The Internet Protocol (IP)

The network layer is modelled according to the Internet Protocol (IP). IP is designed for use in interconnected systems of packet-switched computer communication networks. IP headers contain 32-bit addresses which identify the sending and receiving hosts. These addresses are used by intermediate routers (nodes) to select a path through the network for the packet. IP also provides services, such as fragmentation/reassembly and Packet timeout control. In our simulation, every packet is routed independently without setting up the connection. The implementation of IP in GlomoSim is based on RFC791 [43].

4.1.3 Workload Model

In general, an Ad Hoc network is a kind of temporary network, where every node could work as the packet source sender sometime and be chosen as the packet relaying node or destination at another time. In order to simulate such a traffic mode, pairs of mobile nodes are chosen randomly as the CBR servers and clients separately in our experiments. CBR streams could start at any simulation time and last within the period [1s, 30s] randomly, and the packet inter-arrival time (CBR interval) is set from $1s/packet$ to $0.01s/packet$. The smaller the value of the CBR interval is, the heavier workload the network has. The total number number of CBR streams is 300 when the simulation time is 1500 seconds, and 600 when the simulation time is 3000 seconds.

4.1.4 Mobility Model

The node mobility is decided by the Random Waypoint model [44]. In this model, a mobile node begins by staying in one location for a certain period of time, called the pause time. After that, the node chooses a destination randomly in the simulated region and a speed randomly from a configurable range $[minspeed, maxspeed]$. Then, that node travels toward the newly chosen destination at the selected speed. Upon arrival, the node pauses for "pause time" before repeating the same process. In this model, the pause time acts as the degree of mobility in a simulation: longer pause time amounts to more mobile nodes being stationary for more of the simulation time.

In our experiments, a rectangular space $(1200m \times 800m)$ is used as the simulation region, which forces the usage of longer routes between nodes than would occur in a square space with equal node density. The node speed is within [0,10] meters/s and 8 different values of pause time (0s, 30s, 60s, 120s, 240s, 480s, 960s and 1500s) are used.

4.1.5 Energy Consumption Model

Our energy consumption model is based on Lucent IEEE802.11 Wavelan [45]. In this model, the card specifications for transmission and reception are TX:(5V, 285mA) and RX:(5V, 185mA) respectively. The energy consumption during the idle and sleep time is ignored in our protocol and in the experiments.

4.1.6 Summary

In general, 100 mobile nodes are placed in a 1200×800 rectangle area. The nodes are initially distributed uniformly. They move according to the Random Waypoint model. Source-destination pairs are chosen at random and the source generates CBR traffic (packets) for a period of time. Each packet is routed according to the protocol under study (e.g. PAWF, GPSR AODV or LEAR). The loss of signal at the physical layer and the function of the MAC layer are modelled as well. Furthermore, as nodes transmit and receive packets, their battery level is updated correspondingly, based on the Lucent IEEE 802.11 Wavelan energy consumption specifications.

4.2 Simulation Metrics

We measure the following performance metrics under different network scenarios and energy situations: packet delivery ratio, average number of hops per packet, average delay per packet, average energy consumption per packet, routing overhead, the COV of nodes' energy consumption and workload, network lifetime in the energy-scarce network and the total number of nodes reloaded and reloaded alive.

Packet Delivery Ratio:

The packet delivery ratio is the ratio between the total number of packets successfully reaching the destinations and the total number of packets sent by the sources. Packet delivery ratio is important because it describes the loss rate which in turn affects the maximum

throughput that the network can support. This metric characterizes both the completeness and correctness of the routing protocol.

Average Number of Hops per Packet:

Average number of hops per packet is the average number of nodes (the source sender or intermediate nodes) a packet passing through before it successfully reaches the destination. This metric is an important measure of routing efficiency

Average Delay per Packet

Average delay per packet is the average period of time that packets take to be delivered from the source to the destination. This is an accumulated value of the processing and queuing delay in from the source to the destination. This metric is also used to evaluate the routing efficiency, especially under different network loads.

Average Energy Consumption per Packet

Average energy consumption per packet is defined as the ratio of total energy consumed by all nodes over total packet delivered successfully by all nodes. This is a power-aware evaluation measure and it is affected by the average number of hops and the routing overhead.

Routing Overhead

Routing overhead is measured by the total number of routing control messages transmitted by all the nodes. Routing overhead measures the scalability of a routing protocol, the degree to which it functions in congested or low-bandwidth environments, and its efficiency in terms of consuming node battery. Protocols that send large numbers of routing control packets can also increase the probability of packet collisions and may delay data packets in the network transmission.

Network Lifetime

The lifetime of the network is described as the minimum battery lifetime over all nodes during the simulation. In order to maximize the lifetime of the network, usually the traffic should be routed such that the energy consumption is balanced among the nodes in proportion to their energy reserves. So this is a metric to evaluate the power-awareness ability of PAWF.

Workload of A Node

The workload of a node is calculated by the total number of packets being transmitted and received by the node. This is a metric for the individual node and used to evaluate the workload balance and energy consumption balance of nodes.

COV

COV stands for the Coefficient Of Variation, which is the ratio of the standard deviation to the average mean. The coefficient of variation is a measure of relative dispersion of items to the mean. A knowledge of relative variation is valuable in evaluating experiments. In our simulations, we calculate the COV of nodes' energy consumption and nodes' handled workload.

4.3 Experiment Design

In our evaluation, we compare the performance of PAWF with GPSR AODV, and LEAR, with different node movement patterns and network loads. Experiments are mainly designed and implemented in three types of energy scenarios: full energy, scarcity of energy and scarcity of energy with battery reloading.

In the full energy scenario, every node has a battery with full capacity initially. So no nodes turn off due to the scarcity of energy. This experiment is to evaluate the routing performance of PAWF in a regular network scenario and the ability to balance the energy

usage. When the energy is scarce, the initial energy value of some nodes is not enough to support the load during the whole simulation period, and nodes with low initial energy values or those being over-used could be dead earlier and affect the network performance. Therefore, we design the second type of experiment to test PAWF's ability to balance the energy consumption, avoid the over-usage of nodes and increase the network lifetime. Finally, in the third set of experiments, we assume every mobile node has a backup battery. So nodes in our experiments could be reloaded when the battery level is low or running out. This experiment is to further test PAWF's ability to avoid overusing low-battery nodes and improve the performance of routing in energy-scarce situations.

In order to measure metrics in a fair way, simulations are mainly made by combining 4 routing protocols, 8 movement patterns (8 different values of pause time) and 3 energy scenarios. Other experiments are also implemented with different traffic loads. All the experiments are repeated with 10 different random seed values and different random network topologies. In the energy-scarce situation, 10 different sets of energy initial values are used to obtain the average experimental results. Moreover, the confidence interval values with 95% confidence level are also calculated to verify the difference of experimental results among the routing protocols.

4.4 The Implementation of AODV, LEAR and GPSR

4.4.1 The Implementation of AODV

The Ad hoc On Demand Distance Vector (AODV)[4] routing protocol is a source-initiated on-demand routing protocol, which builds routes between nodes only as desired by source nodes. It has a route discovery mechanism that sends route request messages to discover paths. Nodes maintains these routes as long as they are needed by the sources. The implementation of AODV in GlomoSim follows the specification of AODV Internet Draft Version 3. The parameter settings governing AODV's operation are shown in Ta-

ble 4.2. $NET_DIAMETER$ measures the maximum possible number of hops between two nodes in the network. $NODE_TRAVERSAL_TIME$ is a conservative estimate of the average one hop traversal time for packets. AODV waits $RREP_WAIT_TIME$ before retransmitting the route request. When a node receives a route request and finds another request with the same source IP address and a broadcast ID field was received within the last $BCAST_ID_SAVE$ period, the newly received request will be discarded. To protect against using stale paths, the active route is controlled by the timeout parameter $ACTIVE_ROUTE_TIMEOUT$. Each node has the same value of $MY_ROUTE_TIMEOUT$, which could be used in the route reply message. To prevent unnecessary network-wide broadcasts of route requests, a TTL scheme is also set.

We compare PAWF with AODV in order to analyze the difference of routing performance between position-based routing and topology-based routing.

NET_DIAMETER	35
NODE_TRAVERSAL_TIME	40 MILLI_SECONDS
RREP_WAIT_TIME	3 * NODE_TRAVERSAL_TIME * NET_DIAMETER / 2
BCAST_ID_SAVE	30 SECOND
BROADCAST_JITTER	10 MILLI_SECONDS
RREQ_RETRIES	2
ACTIVE_ROUTE_TIMEOUT	10 SECONDS
MY_ROUTE_TIMEOUT	2 * ACTIVE_ROUTE_TIMEOUT
REV_ROUTE_TIMEOUT	RREP_WAIT_TIME
TTL_START	1
TTL_INCREMENT	2
TTL_THRESHOLD	7

Table 4.2: AODV parameters used in simulation

4.4.2 The Implementation of LEAR

The Local Energy-Aware Routing, LEAR, algorithm is an on-demand topology-based power-aware routing algorithm based on Dynamic Source Routing protocol (DSR). The design goal of LEAR is to balance the battery usage of nodes and to increase the lifetime of the

network, which is similar to the goal of PAWF.

DSR is a routing protocol designed for MANETs. It is a source-routing protocol and composed of two main mechanisms: route discovery and route maintenance. In the process of route discovery, a route request is send from a node S to a node D by broadcast, only when S attempts to send a packet to D and has no available route to D in its route cache. So the node S does not always know a route to D and the route request proceeds completely on-demand to reduce the routing overhead. Intermediate nodes piggyback their ID into the source route included in the route request message and relay that route request by broadcast, if they do not know an available path to D. When the route request reaches D or some intermediate node which knows the route to D (by checking its route cache), a route reply is unicasted to S with the complete path from S to D. Nodes could receive the same route request more than one time, but only the first one will be handled. Node S could also receive multiple routing replies. The first arrival route is used immediately. If in the following routing replies, a shorter path from S to D is included, the new route will be used instead of the old one. If S can not receive a route reply after a period time, the route request will be resent until a path to D is finally discovered. Route maintenance is the mechanism by which node S is able to detect, while using a source route to D, if the network topology has changed such that it can no longer use its route to D because a link along the route no longer works. When route maintenance indicates a source route is broken, S can attempt to use any other route it happens to know to D, or can invoke route discovery again to find a new route for subsequent packets to D. Route maintenance is used only when S is actually sending packets to D.

LEAR is designed based on the route discovery and route maintenance mechanism of DSR. But in the route discovery process, LEAR adds some power-aware metrics to balance the battery usage of nodes. First, an adjustable battery threshold value TH_r is defined for each node. During the route discovery process, if a node receives a route request, and its

battery level is greater than TH_r, it handles that route request like DSR. Otherwise, it drops the route request, not relaying it or sending a route reply. With this method, the destination will receive a route request only when all intermediate nodes along the route have good battery levels. Thus, the over-usage of low-battery nodes is avoided and the battery usage of nodes is balanced. In order to adjust the value of TH_r for each node, an adjustment factor $d\,(0 < d < 1)$ is defined. After a node A drops a route request, if that route request is resent and node A receives it again, A will reduce its TH_r by d (that is $TH_r = TH_r * (1 - d)$) and use the new TH_r value for the next step operation.

There are two main problems that LEAR has to handle: how to avoid the cascading effect and how to use the route cache.

The cascading effect will occur when multiple intermediates nodes from S to D have lower battery level than their battery threshold. For example, a path is supposed to exist from $S \rightarrow N_1 \rightarrow N_2 \rightarrow N_3 \rightarrow D$ and the battery levels of node N_1, N_2 and N_3 are all below their TH_r value. If a route request is sent from S to D, it will be dropped by N_1. When the request is resent, N_1 will reduce its TH_r value and could relay the request. But the route request is now dropped by node N_2. For the same reason, the third resent request will be dropped by N_3. Therefore, in order to let D receive a route request from S, the request has to be sent at least four times. In order to handle this problem, when node N_1 drops the route request, it forwards a $DROP_ROUTE_REQ$ to D, with the source address, the destination address and the sequence number of the route request. Then, subsequent nodes (N_2 and N_3) closer to D now know that a request message was dropped and lower their threshold values when they receive the retransmitted route request. In this way, the second route request message is possible to reach D and the cascading problem is solved.

In DSR, if the intermediate node receives a route request and finds an available route in its route cache, it can simply add that route into the source route and reply to the source

node. With LEAR, such a reply can not simply provided, because the intermediate node does not have information on the battery levels of the route from itself to the destination in its route cache. In order to handle this problem, three additional routing-control messages are utilized $ROUTE_CACHE$, $DROP_ROUTE_CACHE$ and $CANCEL_ROUTE_CACHE$. When a node A receives a route request and knows the path to the destination from its route cache, it unicasts a $ROUTE_CACHE$ message to the destination through that path to determine the battery levels of the following nodes along the path. When a node B receives a $ROUTE_CACHE$ message and its battery level is above its TH_r, the $ROUTE_CACHE$ is relayed to the next hop node based on the source route carried. Otherwise, it replies with a $CANCEL_ROUTE_CACHE$ back to the node that started sending the $ROUTE_CACHE$ message (node A, in this case), so that A can drop the corresponding route request and invalidate that entry in its route cache. At the same time, B also unicasts a $DROP_ROUTE_CACHE$ message to the following nodes in the source route carried by $ROUTE_CACHE$. Then, those subsequent nodes can adjust their TH_r by d when receiving the retransmitted route request message. If the destination receives a $ROUTE_CACHE$, it means the path from the sender of the $ROUTE_CACHE$ to the destination satisfies the battery requirement of LEAR. So the destination sends a reply to the sender of the $ROUTE_CACHE$ (node A, in this case) and then A can send a route reply to the source sender of the route request it received. With this method, the route cache of intermediate nodes is able to be used, while ensuring a battery-sufficient path.

The parameter settings governing LEAR's operation are shown in Table 4.3. The parameters for route discovery and maintenance are set based on [11]. Route Discovery is the on-demand process by which nodes actively obtain source routes to destinations to which they are actively attempting to send packets. After each route discovery attempt, the interval between successive route discoveries for this target must be doubled. The minimum back-off period of re-sending the route request is $REQUEST_PERIOD$, and the

maximum of it is $MAX_REQUEST_PERIOD$. The route request will be saved by the request source sender and intermediate nodes for a $FLUSH_INTERVAL$ period. At the same time, the packet waiting for transmission is also buffered in the source sender for a $MAX_PACKET_BUFFER_TIME$ time. Battery threshold TH_r and adjustment factor d are two important parameters, whose values have to be set heuristically. If the initial value of TH_r is too low, every node has a battery level above its TH_r and LEAR is completely reduced to be DSR. If the TH_r value is too high, at the beginning, most nodes have a lower battery level than their TH_r and refuse to join the route recovery and packet transmission, which will induce very low packet delivery ratio. If the d value is set too large, the TH_r value of nodes will be reduced very quickly, which decreases LEAR's power-awareness performance. If the d value is too small, the route requests will be dropped more frequently by low-battery nodes, which increases the routing overhead and decreases the delivery performance. In our experiments with scarce energy, the initial battery levels of nodes follow the normal distribution with $MEAN = 15000mWsec$. Therefore, the battery threshold TH_r of every node is set as 90% of the $MEAN$ value. The adjustment factor d is defined as 0.1 or 0.4 separately, the same values used in [5].

Since PAWF and LEAR have the similar design goal of balancing the battery usage of nodes, the comparison between them is helpful to analyze the effect of different design strategies to the performance of power-aware routing.

REQUEST_PERIOD	500 * MILLI_SECOND
MAX_REQUEST_PERIOD	10 * SECOND
FLUSH_INTERVAL	30 * SECOND
MAX_PACKET_BUFFER_TIME	30 * SECOND
BROADCAST_JITTER	10 * MILLI_SECOND
TH_R	90% of MEAN battery initial values
d	0.1 or 0.4

Table 4.3: LEAR parameters used in simulation

4.4.3 The Implementation of GPSR

The Greedy Perimeter Stateless Routing, GPSR [2], is a pure position-based routing protocol. GPSR uses greedy forwarding to forward packets to nodes that are closer to the destination. When that fails, GPSR works in the recovery mode, using the planar graph face-crossing method. It switches back to the greedy mode, when current node has a shorter distance to the destination than the entrance node of the recovery process. The implementation of GPSR over GlomoSim is based on [2] and GPSR's NS-2 simulation code. The parameter settings governing GPSR's operation are shown in Table 4.4, which are the same as those used in PAWF.

With the comparison between PAWF and GPSR, we verify the ability of PAWF to balance the battery usage of nodes and to prolong the lifetime of the network.

NEIGHBOR_BEACON_INTERVAL	5 SECONDS
NEIGHBOR_ENTRY_LIFETEIM	2 * NEIGHBOR_BEACON_INTERVAL

Table 4.4: GPSR parameters used in simulation

In our simulation, AODV, GPSR, LEAR and PAWF use the same physical layer setting, workload model, mobility model and energy consumption model to ensure the fairness of the performance evaluation.

4.5 Analysis of Experimental Results

For evaluating PAWF, simulation results are obtained by experiments with different energy levels, different traffic loads and different movement patterns of nodes.

4.5.1 Routing Performance with Enough Energy

In the first set of experiments, every node is given a battery with full capacity initially. In this scenario, no nodes turn off due to running out of energy. This is to test the routing

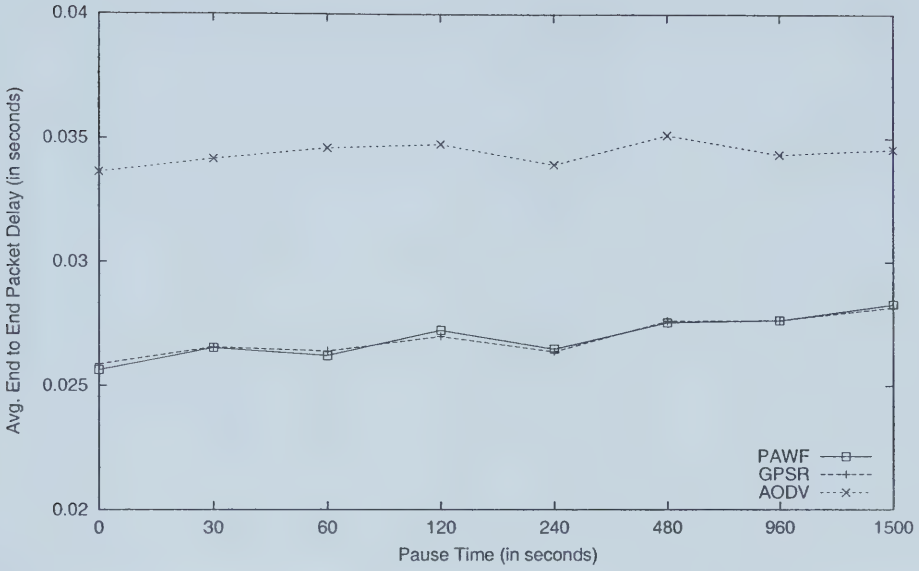


Figure 4.3: Average end-to-end delay per packet : 300 CBRs, CBR interval = 0.2s/packet, full battery initially, simulation time = 1500s

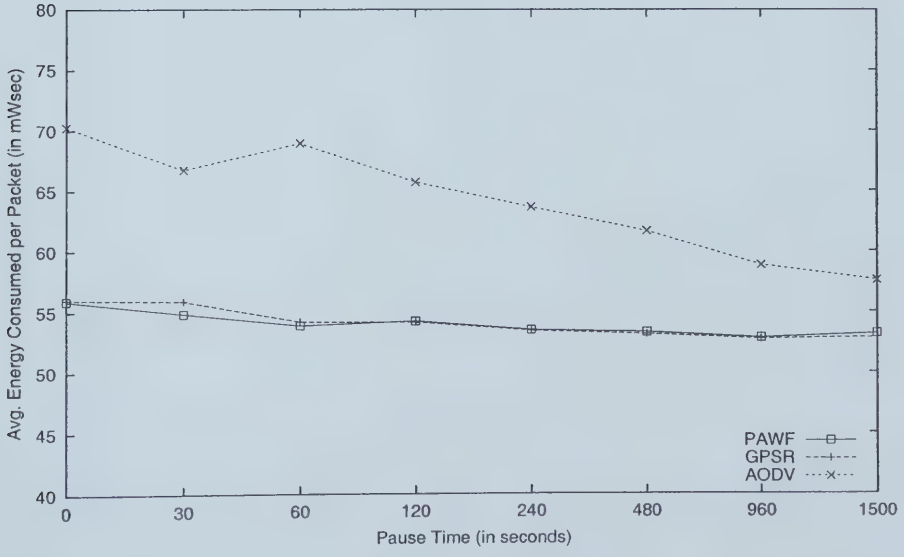


Figure 4.4: Average energy consumed per packet : 300 CBRs, CBR interval = 0.2s/packet, full battery initially, simulation time = 1500s

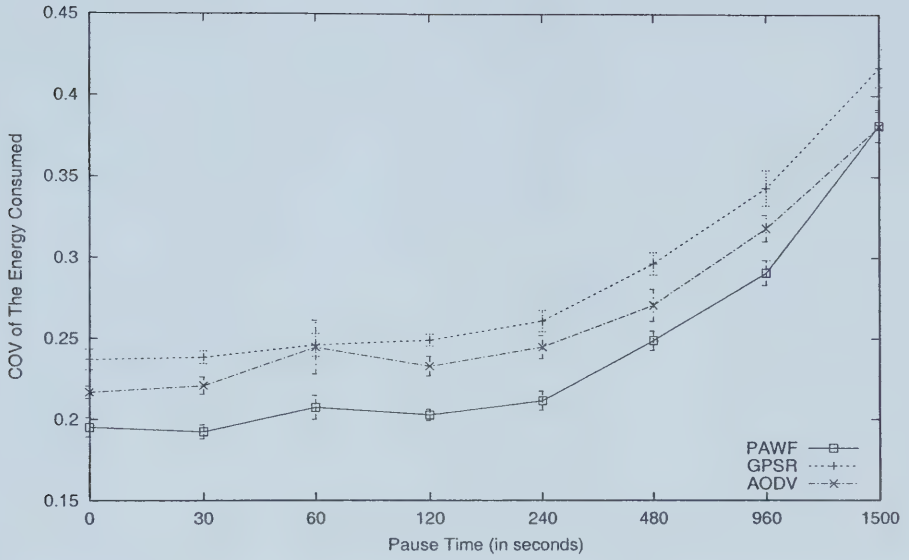


Figure 4.5: COV of energy consumed for nodes: 300 CBRs, CBR interval = 0.2s/packet, full battery initially, simulation time = 1500s

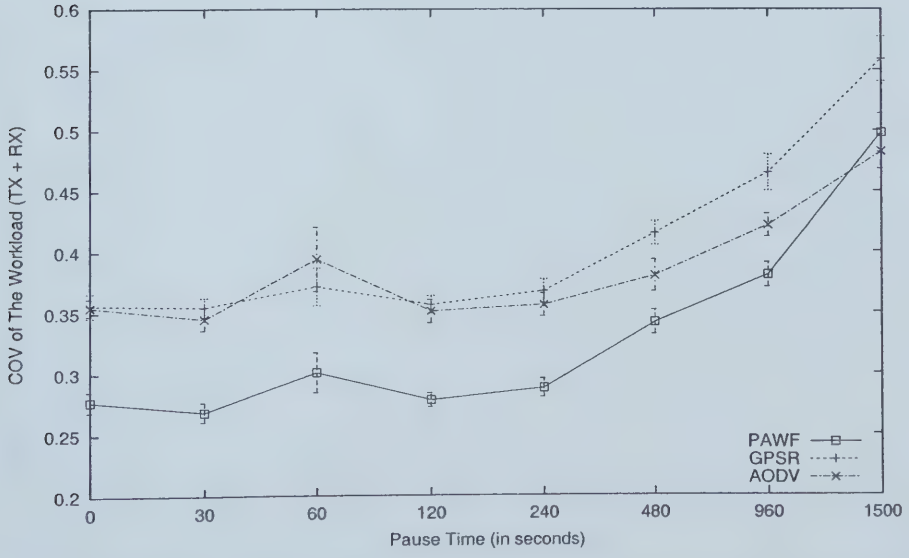


Figure 4.6: COV of the workload for nodes : 300 CBRs, CBR interval = 0.2s/packet, full battery initially, simulation time = 1500s

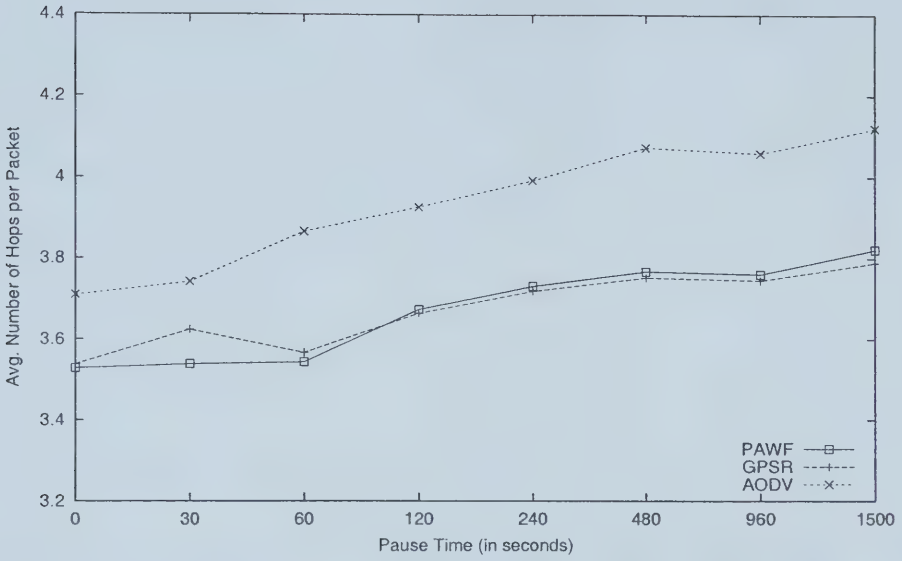


Figure 4.7: Average number of hops per packet : 300 CBRs, CBR interval = 0.2s/packet, full battery initially, simulation time = 1500s

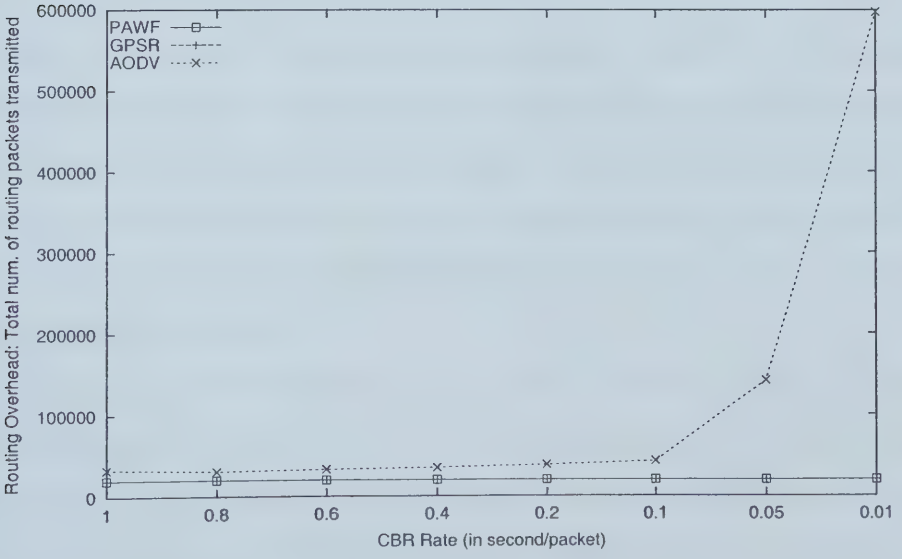


Figure 4.8: Routing overhead on different network workloads: 300 CBRs, Pause = 0s, full battery initially, simulation time = 1500s

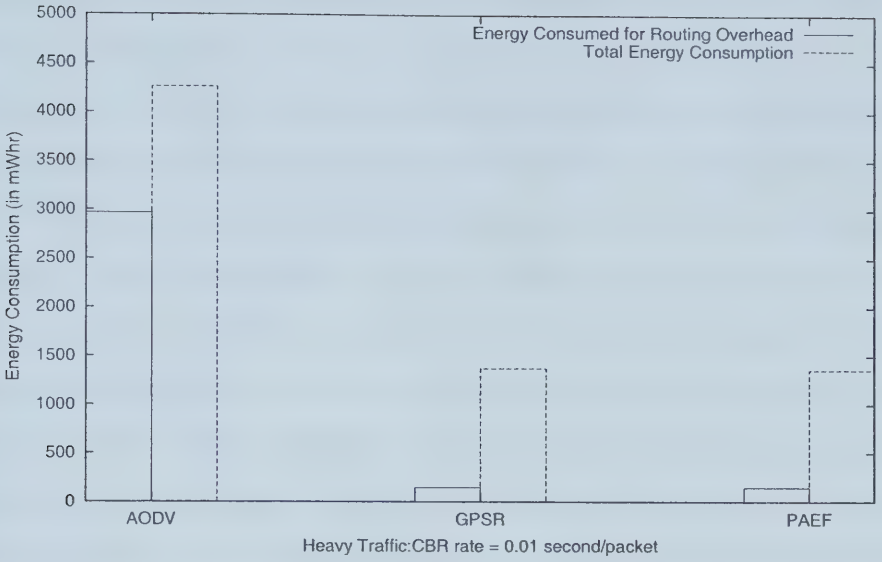


Figure 4.9: Energy consumption for transmitting and receiving routing messages on heavy workload : 300 CBRs, CBR interval = 0.01 s/packet, pause = 0s, full battery initially, simulation time = 1500s

performance of PAWF in a regular network scenario. A total of 300 CBR streams are generated within the 1500-second simulation time.

Figure 4.1 to Figure 4.7 show the routing performance of AODV, GPSR and PAWF with respect to different pause times (that is levels of mobility) for metrics of delivery ratio, routing overhead, average end-to-end delay, average energy consumption per packet, COV of energy consumed, COV of workload and average number of hops per packet respectively. Figure 4.8 and 4.9 show routing overhead and energy consumed for routing overhead with respect to different CBR rates.

From Figure 4.1, PAWF and GPSR have very similar packet delivery ratio, about 97% to 99%. AODV performs a little bit better than the two position-based approaches when node mobility is high (Pause time = 0 - 120 s). Such a result is not surprising. The better delivery ratio of ADOV comes from its routing failure feedback strategy. AODV is a flooding-based routing algorithm, which triggers route requests when there is no valid route existing. If a

node along a route moves, its upstream neighbor notices the move and propagates a link failure notification message to each of its active upstream neighbors to inform them to update that part of the route. Such a feedback propagation continues until reaching the source node of that route. Therefore, AODV could react quicker to the topology changes of the network than position-based ones which mostly depend on periodical beacon broadcasting, and possibly obtains higher delivery ratio when mobile nodes move frequently. However, such a performance improvement is expensive, which induces high routing overhead (Figure 4.2), and consequently increases the energy consumption per packet (Figure 4.4). In fact, position-based routing algorithms are able to increase the delivery ratio by reducing the neighbor broadcast interval or sending a unicast on-demand neighbor request to confirm the availability of the intended next hop node. But these methods will also increase the routing cost. A tradeoff here is to design a hybrid neighbor beacon strategy. Most of the time, only periodical beacons are sent. On-demand beacons or requests are used only for special situations of nodes, such as waking up after long periods of sleep, battery loading/reloading, or when the neighbor table is empty. The usage of on-demand beacons could reduce network/node warming up period and avoid unnecessary packet drops.

With Figure 4.5, the COV of energy consumed per node with PAWF is 20% to 25% lower than GPSR. So PAWF does balance the energy usage of nodes. Consequently, the workload each node handles is also balanced, which is shown in Figure 4.6 (COV of workload). At the same time, with the usage of our asymmetrical weighted forwarding function, PAWF keeps similarly high routing efficiency to GPSR, such as similar packet delivery ratio (Figure 4.1), routing overhead (Figure 4.2 and 4.8), end-to-end delay per packet (Figure 4.3) and average number of hops per packet (Figures 4.7). Therefore, we can conclude that the usage of our power-aware weighted forwarding function does not affect the efficiency of routing and PAWF has the ability to find the global near-optimal energy-aware path by the local optimal choices.

With the comparison of AODV, PAWF also has several advantages. First, the routing overhead (for packet forwarding) of PAWF is lower than AODV, which is shown in Figure 4.2 and 4.8. Moreover, the routing overhead of PAWF is constant, not relative to the change of node mobility and network traffic load. For AODV, it works the other way around. From Figure 4.2, AODV's routing overhead increases greatly with the increase of node mobility. In fact, for AODV, the overhead when pause = 0s is double that when pause = 1500s. The routing overhead of AODV also increases with the increase of network traffic. In Figure 4.8, the routing overhead of AODV increases 2000% higher when the CBR rate decreases from 1s/packet to 0.01s/packet. When the traffic load is heavy, packet routing with AODV is much more expensive than with PAWF, because a large amount of energy is consumed on routing overhead by AODV (shown in Figure 4.9). At the same time, heavy routing overhead takes more system resources, increases the traffic load of the network and induces higher end-to-end packet delay than PAWF (shown in Figure 4.3). Based on above results, PAWF is more energy-efficient than AODV in high mobility or heavy-loaded networks. It should be noted that, PAWF's routing performance is evaluated based on the assumption of accurate destination positions. If the destination is not fixed or not predictable, a location service is needed for the source sender. Then, the performance of PAWF will be affect by the accuracy of the location service to a corresponding level and the total routing overhead of PAWF will be increased correspondingly.

In MANETs, nodes keep moving in the network area. For PAWF and GPSR, the next hop node is chosen based on the neighbor table, which is mainly updated periodically. Therefore, it could be possible that the delivery fails due to the stale position information of nodes or stale neighbor entries. We show, in Table 4.5, the delivery performance of three routing protocols and the delivery failures due to different reasons. First, we measured the delivery failures due to the stale position information of the packet destination. This type of failures is induced mainly by two situations: stale neighbor entry for the final hop

	PAWF	GPSR	AODV
Number of packets prepared to be delivered	23535	23535	23535
Successful delivery ratio	97.02%	96.96%	98.46%
Number of delivery failures due to stale position information of the destination	188 (0.80%)	186 (0.79%)	
Number of delivery failures due to stale position information of overall nodes through the path	660 (2.80%)	671 (2.85%)	
Number of delivery failures due to broken routes			332 (1.41%)
Number of delivery failure due to other reasons	40 (0.18%)	44 (0.19%)	30 (0.13%)

Table 4.5: Delivery performance analysis: 300 CBRs, CBR interval = 0.2s/packet, pause time = 0s, full battery initially, simulation time = 1500s

or stale direction to the destination. Stale neighbor entry for the final hop means that, for the final relaying node, it finds the destination in its neighbor table and then believes it can transmit the packet directly to it. But it could be possible that the packet destination node had moved out the radio range of that node after the last neighbor table update. In this case, the packet can not reach the destination and the delivery fails. Stale direction to the destination also come from the mobility of the destination. For PAWF and GPSR, if the destination of the packet is not fixed, the source sender of the packet piggybacks the destination position into the packet, which is used by the following nodes without change. Therefore, if the destination moves very fast and nodes still try to transmit the packet to the stale piggybacked position of the destination, the packet delivery could fail. In our experiments, the second type of delivery failures rarely occur, because the average end-to-end packet delay is very short, less than 0.03 second and the speed of nodes is not high, up to 10 m/s. So during transmitting the packet from the source to the destination, the movement of the destination is ignorable and has very small effect on the delivery process. Of course, this simulation results are based on the accurate destination position at the time of the source node transmitting. The effect of the destination position to the delivery performance could be higher, if a location service is used, which is not up to date.

Moreover, when the traffic load is heavy which increases the end-to-end delay, or when the speed of nodes is very high, the accuracy of the destination position could affect the delivery performance more. In this situation, we can piggyback the destination position with a time stamp at the source sender. Then, for any intermediate nodes which happen to have a fresher position of the destination, they could update the stale information with the new one in the packet. Totally, we measured that about 0.8% packets are delivered unsuccessfully due to the above two situations of destinations. Based on the process of pure position-based routing, the delivery failures due to the stale neighbor entry (or position information of nodes) could happen at every hop. For GPSR and PAWF, totally about 2.8% packets are delivered unsuccessfully for stale neighbor entries of overall nodes through the path, which is measured with $pausetime = 0s$ and the mobility of nodes is relative high. There is also a very small part of delivery failures happening due to some other reasons, such as radio layer collision or the effect of noise.

For AODV, the packet delivery does not depend on the position of the destination. It picks up the route from its route cache or discovers it by an on-demand route request if no route is currently available. If the path is freshly discovered by the route request, the packet can usually be delivered successfully to the destination through that path. But if the cached route is used, which could be stored in the cache for a period of time, it is possible that some nodes through that path have moved to some other places and then the route is broken. For AODV, totally 1.41% packets are delivered unsuccessfully due to broken routes. Also, a very small part of delivery failure is due to other reasons.

In this set of experiments, we do not present the results of LEAR. Since every node initially has full battery and all nodes's battery levels are above their TH_r threshold, LEAR protocol, in this case, is reduced to the pure DSR protocol, which is a source-routing on-demand approach and has similar routing performance and properties to AODV.

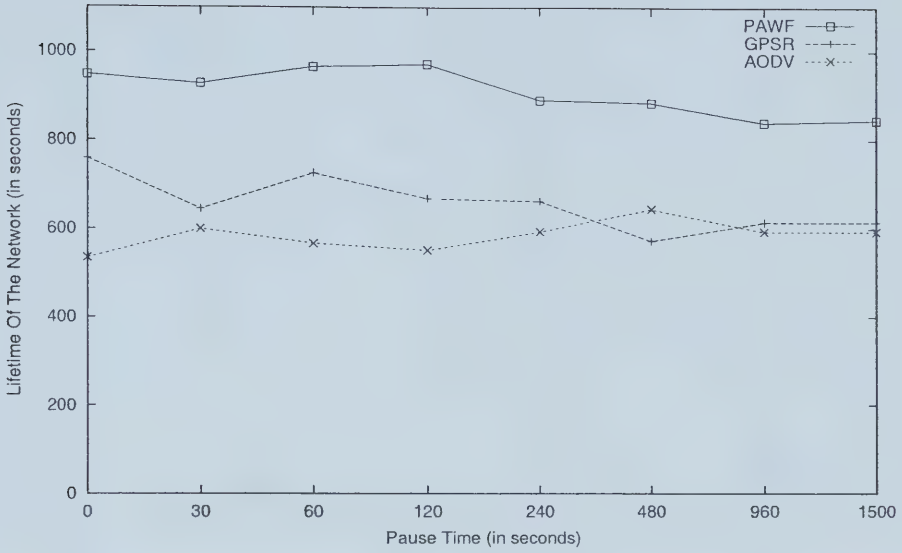


Figure 4.10: Lifetime of the network under different movement patterns: 300 CBRs, CBR interval = 0.2s/packet, limited energy initially, simulation time = 1500s

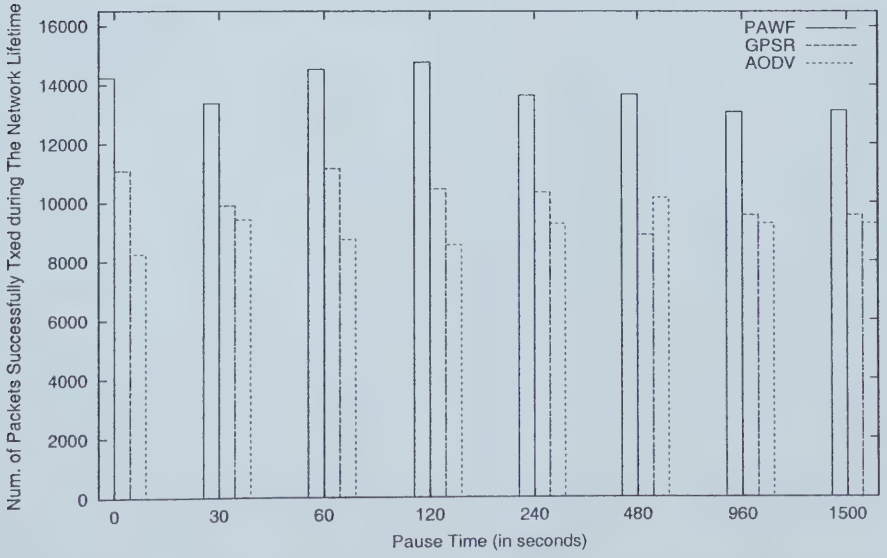


Figure 4.11: Number of packets delivered during the network lifetime: 300 CBRs, CBR interval = 0.2s/packet, limited energy initially, simulation time = 1500s

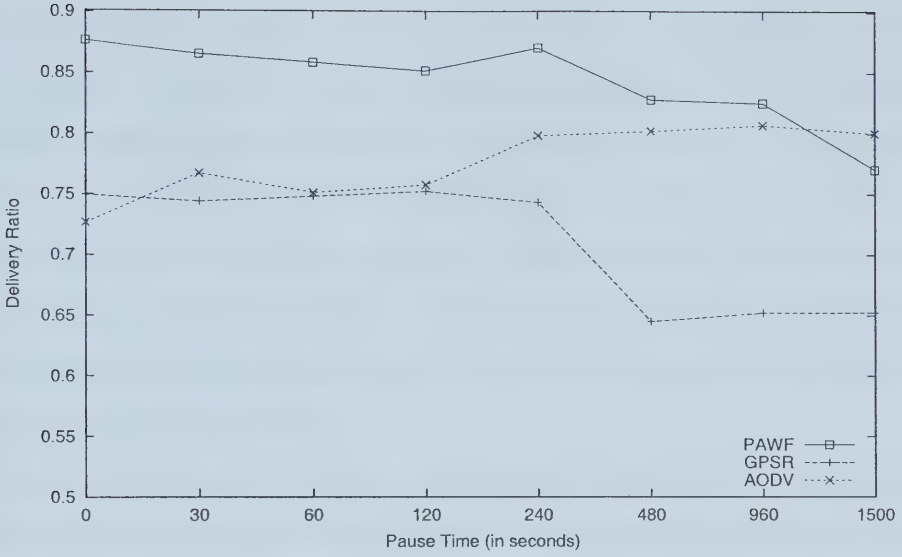


Figure 4.12: Packet delivery ratio during the simulation time: 300 CBRs, CBR interval = 0.2s/packet, limited energy initially, simulation time = 1500s

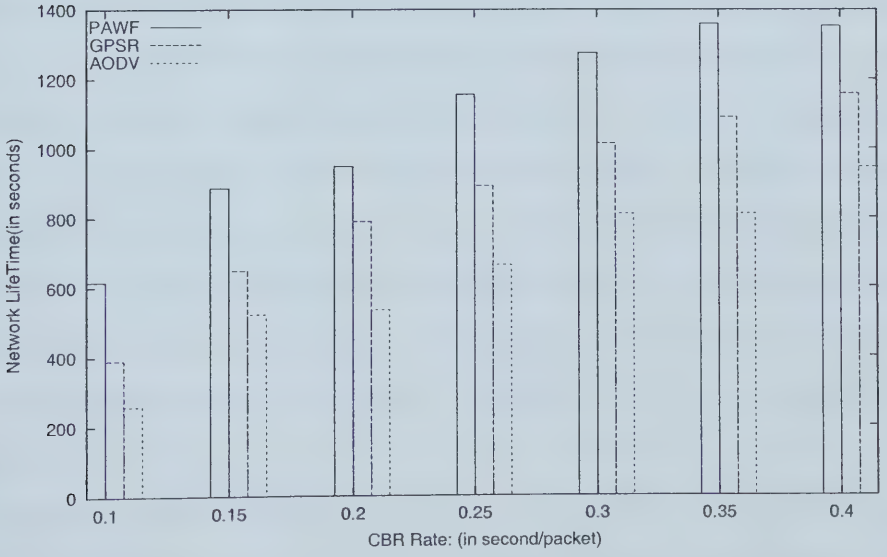


Figure 4.13: Lifetime of the network on different network workloads: 300 CBRs, pause = 0s, limited energy initially, simulation time = 1500s

4.5.2 Routing Performance with Insufficient Energy

In the second set of experiments, every node is only given a small part of its full energy. This is to test PAWF's ability to balance the energy consumption, avoid the over-usage of nodes and increase the network lifetime. The initial energy value of each node follows a normal distribution with $mean = 15000mWsec$ and $STDEV = 3000mWsec$. The simulation time is still 1500 seconds and a total of 300 CBR streams are generated. Since the initial energy value of some nodes are not enough to support the whole simulation period, nodes with low initial energy values or those being over-used could be dead earlier and affect the network performance.

Figure 4.10 to Figure 4.12 shows the experimental results for metrics of network life time, number of packet delivered during the network lifetime and packet delivery ratio over the simulation time respectively, with respect to changing pause time. Figure 4.13 shows the lifetime of the network with respect to changing CBR rate.

In energy insufficient situations, how to avoid the over-usage of low-battery nodes is crucial to prolong the network lifetime. From Figure 4.10, PAWF keeps 20% to 40% longer lifetime and 15% to 20% higher packet delivery ratio than GPSR. Compared with AODV, the lifetime increased by PAWF is more obvious, about 30% to 80%. Therefore, based on Figure 4.11, the number of packets delivered by PAWF is 20% to 30% higher than that by AODV and GPSR during the network lifetime. When node mobility is high (pause = 0-120s), PAWF has 10% to 15% higher delivery ratio than AODV during the simulation time. With decreasing the mobility of nodes, the delivery ratio of AODV increases. When nodes are static/near-static (pause = 480-1500s), PAWF and AODV have very similar delivery performance.

The increase of lifetime by PAWF comes from the usage of our power-aware weighted forwarding function, which always chooses nodes with relatively enough energy to be the next hop node when possible. Since energy-sufficient nodes take more routing work, the

low-battery nodes keep alive for a longer time and the integrity of the network is also prolonged by PAWF.

When the number of dead nodes increases, packets are usually dropped in one of two situations:(1) The network is partitioned and no physical path exists between source and destination; (2) the destination runs out of energy and quits. With PAWF, nodes are used in a more energy-balanced way and the connectivity of the network is kept for a longer time. At the same time, compared with GPSR, PAWF achieves power-awareness without requesting any extra routing control messages. So PAWF allows more packets to be successfully transmitted than GPSR with limited energy supplied.

The delivery ratio of AODV increases with the decrease of mobility, because the corresponding routing overhead of AODV is reduced. The reduction of routing overhead leads to an amount of energy saved and relatively more energy is used for data delivery. Moreover, compared with position-based approaches, AODV has stronger ability to detect physical disconnection between source and destination pairs. Unlike PAWF and GPSR, which keep transmitted data packets until no next hop is available, AODV usually drops the packet at the source sender side and saves the energy of data transmission consequently.

Experiments are also made under different network workloads. The experimental results are shown in Figure 4.13, where different CBR intervals are used from 0.4s/Packet to 0.1s/Packet. We find PAWF keeps 20% to 40% longer network lifetime than GPSR and 40% to 80% longer than AODV. Therefore, in summary, we can conclude that PAWF achieves much longer network lifetime than GPSR and AODV, and achieves excellent packet delivery performance in high mobility networks. PAWF has a strong ability to balance the usage of nodes based on the battery situation and avoid the over-usage of nodes. This property is upheld under different network workloads. With the global routing discovery process and decreased routing overhead, AODV has similar delivery performance to PAWF in the static/near static network environment.

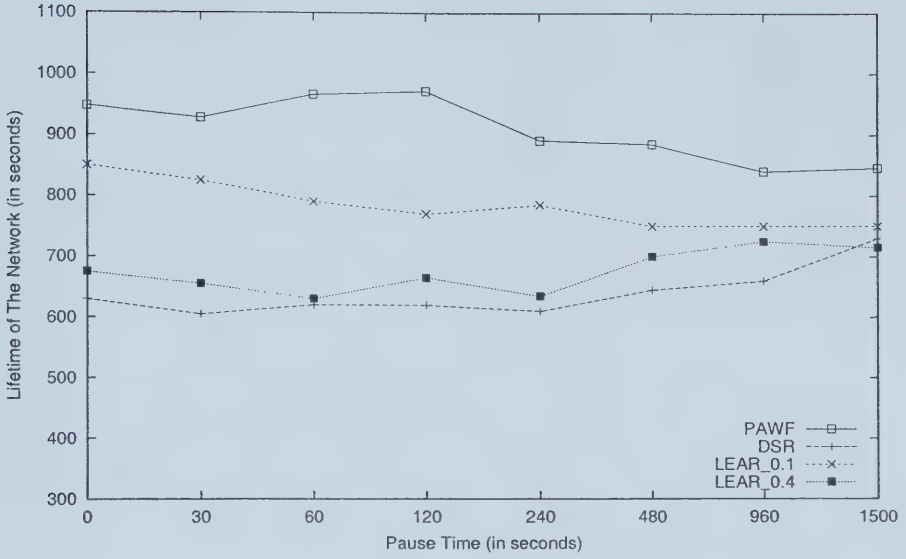


Figure 4.14: Lifetime of the network under different movement patterns: 300 CBRs, CBR interval = 0.2s/packet, limited energy initially, simulation time = 1500s

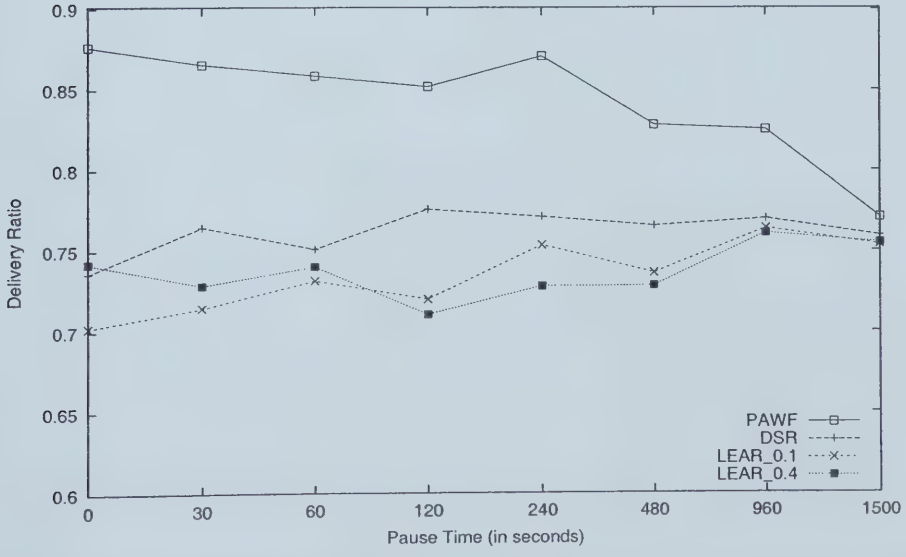


Figure 4.15: Packet delivery ratio during the simulation time: 300 CBRs, CBR interval = 0.2s/packet, limited energy initially, simulation time = 1500s

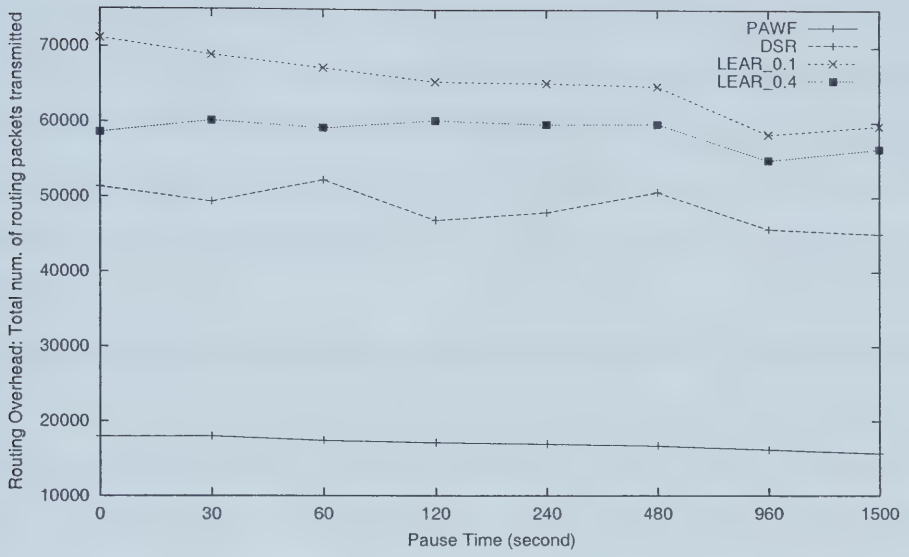


Figure 4.16: Routing overhead during the simulation time: 300 CBRs, CBR interval = 0.2s/packet, limited energy initially, simulation time = 1500s

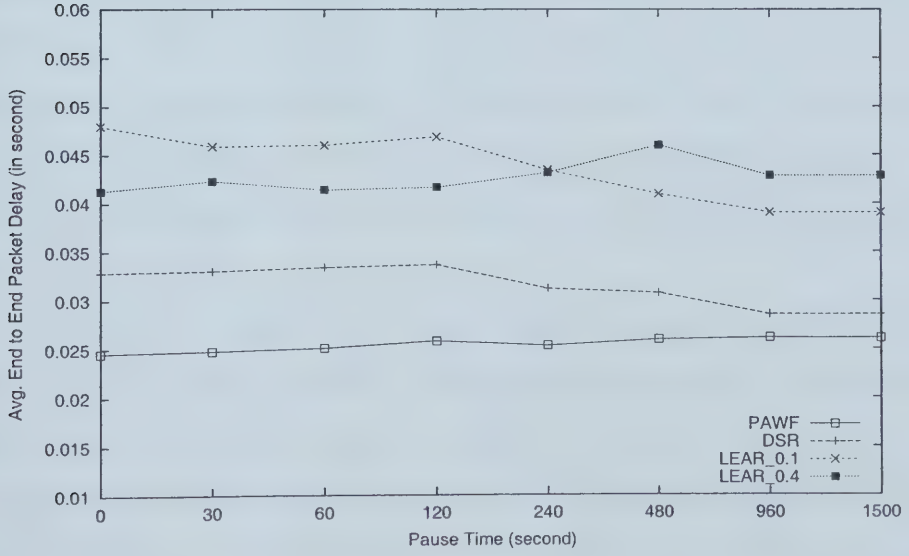


Figure 4.17: Average end-to-end delay per packet during the simulation time: 300 CBRs, CBR interval = 0.2s/packet, limited energy initially, simulation time = 1500s

Besides comparing PAWF with two non-power-aware approaches, AODV and GPSR, an on-demand power-aware routing protocol, LEAR, is also implemented and the experiment results are shown as follows. Figure 4.14 shows the network lifetime achieved by different protocols. Figure 4.15 shows packet delivery ratio during the simulation time. Figure 4.16 and 4.17 show the routing overhead and end-to-end delay of packets during the simulation time. All experiment results are presented with respect to changing pause time of nodes.

LEAR is a power-aware routing protocol designed based on DSR. It has a similar design goal to PAWF to balance the energy usage of nodes. With LEAR, only nodes with enough energy are willing to take part in the route discovery process. So for the packet source sender, the path it discovers usually consists of nodes with good battery status. With this method, the low-battery nodes are usually allocated with light-traffic so as to balance the battery usage of nodes and increase the lifetime of the network. In order to achieve this goal, an adjustable battery threshold TH_r and an adjustment factor d is defined. Four additional routing-control messages are also used to handle the cascading effect of routing and ensure the correct usage of route cache of intermediate nodes (this is described in Section 4.4.2). In our experiments, the TH_r value is set as $90\% * MEAN$ value of initial battery levels of all nodes. d is set as 0.1 and 0.4 separately. After dropping a route request and receiving the retransmission of that route request, nodes reduce their TH_r by d (that is $TH_r = TH_r * (1 - d)$).

From Figure 4.14, compared with DSR, LEAR_0.1 does increase the lifetime of the network, especially when node mobility is high (small pause time value). For LEAR_0.4, such increase of lifetime is not as great as LEAR_0.1. The reason is, with $d = 0.4$, nodes reduce their TH_r values by a relatively bigger value after receiving retransmission of route requests. In this way, low-battery nodes will reduce their TH_r value to be low very soon and have to rejoin the data transmission work after then. So, a large d value has weaker

ability to avoid the overusage of scarce-battery nodes than a small d value, and decrease the power-awareness of LEAR.

We know LEAR is a protocol based on DSR plus power-aware metrics. From Figure 4.16, both LEAR_0.1 and LEAR_0.4 have larger routing overhead than DSR. That is because LEAR has to use several types of extra routing control messages to achieve the power-awareness. In order to handle the cascading effect, nodes have to forward $DROP_ROUTE_REQ$ messages after dropping route requests. In order to ensure the usage of route cache, $ROUTE_CACHE$, $DROP_ROUTE_CACHE$ and $CANCEL_ROUTE_CACHE$ messages have to be used to check the battery level of the path in the route cache. For this reason, both of the two LEAR protocols have 20% to 40% higher routing overhead than DSR. LEAR_0.1, which has a small d value, drops route requests more frequently and the correspondingly increased overhead is larger than LEAR_0.4. In an energy-scarce environment, the energy of nodes is limited. Since LEAR spends a larger amount of energy on routing overhead than DSR, a relative small part of energy is left for data transmission. That is why LEAR does not increase the packet delivery ratio to a corresponding level, even though it increases the lifetime of the network (shown in Figure 4.15)

Compared with LEAR, PAWF has some advantages. From Figure 4.16, PAWF has much lower forwarding routing overhead than DSR and LEAR, because PAWF does not broadcast route requests for the route discovery. Moreover, compared with regular position-based routing, PAWF achieves power-awareness without adding any extra routing overhead. Since the routing decision is made hop by hop, only the strict neighbor information is required for PAWF routing. So PAWF is able to balance the battery usage of nodes, increase the lifetime of the network and, at the same time, improve the packet delivery performance in scarce-energy networks. Another advantage of PAWF is it has smaller end-to-end packet delay than LEAR. For regular topology-based on-demand protocols, such as

DSR, the delay of the first packet transmission is relative high due to the route recovery process. For LEAR, such delay is higher, because it can not use the cached route to the destination in the intermediate nodes. In order to check the battery status of the cached route, the intermediate node has to send a $ROUTE_CACHE$ to the destination and wait for the reply or a $CANCEL_ROUTE_CACHE$ for the next-step decision. This strategy increases the round-trip time of route discovery and consequently increases the end-to-end delay of packets. Moreover, with LEAR, the low-battery nodes usually drop the first route request. So the packet source senders have to retransmit the route request and wait for an available path to the destination for a longer time. With PAWF, no route recovery is needed and end-to-end delay is reduced as much as possible. So PAWF has shorter end-to-end packet delay than DSR and LEAR. This property of PAWF can benefit more in large scale networks where the path from the source to the destination is longer.

In summary, both PAWF and LEAR are energy-balanced routing protocols for MANET. LEAR's power-awareness is achieved by extra routing cost and affected by the size or topology changes of the network. PAWF is a complete power-aware local-routing approach, which keeps low and constant overhead, low end-to-end delay and no extra routing control messages are needed to achieve the power-awareness.

4.5.3 Routing Performance with Inefficient Energy and Battery Reloading

In a real application, it could be possible that the users of mobile devices own a backup battery or a fast charger. When a mobile node's battery situation is critical, that node could have the backup battery reloaded and put into operation again. So this set of experiments are to further test PAWF's ability to avoid overusing low-battery nodes and improve the performance of routing in energy-scarce situations. In our research, the initial energy value of every node is set to the same as that in the experiments of insufficient energy, but we assume every mobile node has a backup battery. The critical energy threshold is set as

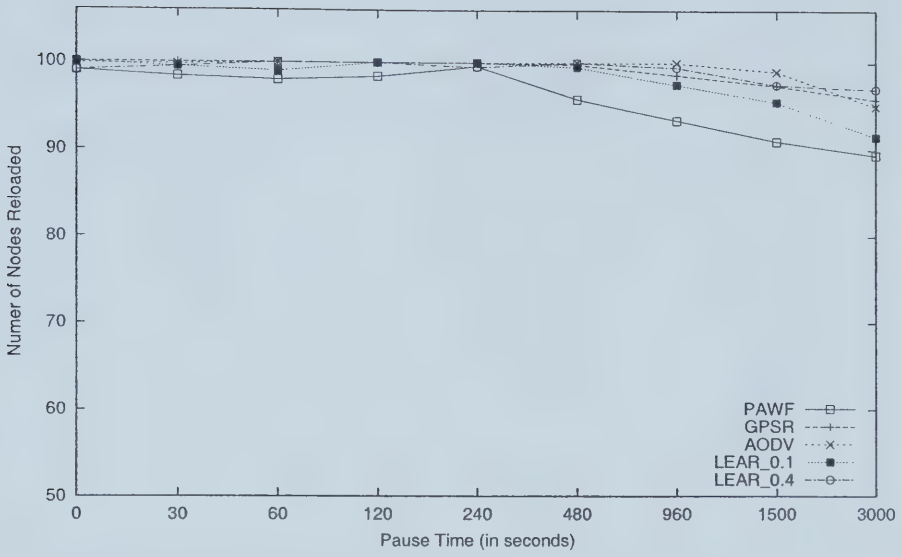


Figure 4.18: Number of nodes being reloaded: 600 CBRs, CBR interval = 0.2s/packet, limited energy initially, battery warning threshold = 5000mWsec, battery reloading period = 1-10mins, simulation time = 3000s

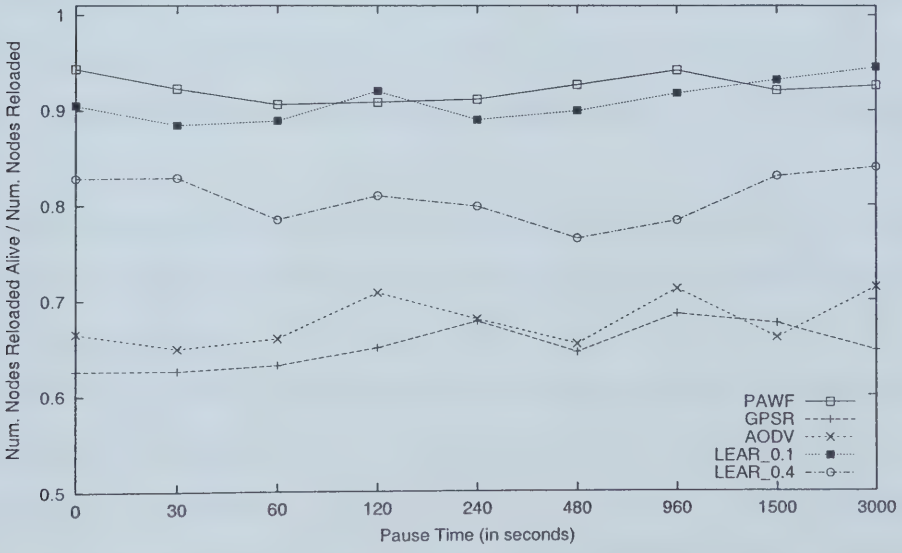


Figure 4.19: Number of nodes being reloaded when alive: 600 CBRs, CBR interval = 0.2s/-packet, limited energy initially, battery warning threshold = 5000mWsec, battery reloading period = 1-10mins, simulation time = 3000s

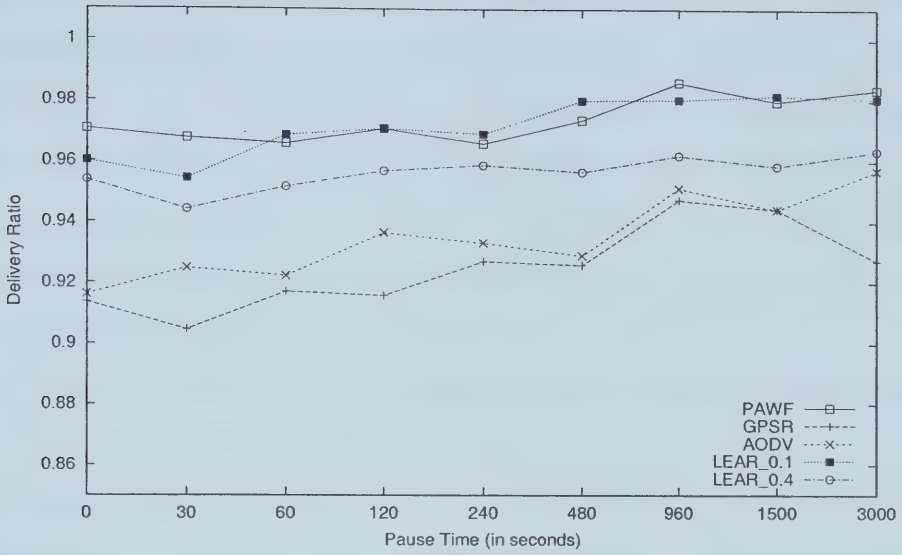


Figure 4.20: Packet Delivery Ratio: 600 CBRs, CBR interval = 0.2s/packet, limited energy initially, battery warning threshold = 5000mWsec, battery reloading period = 1-10mins, simulation time = 3000s

5000mWsec. When a mobile node has lower battery level than that threshold value, a new battery will be reloaded within [1, 10] minutes. Before being reloaded, that node still keeps working as long as it has energy, but could possibly turn off due to running out of energy. The simulation time is increased to 3000 seconds and a total of 600 CBR streams are generated.

In Figure 4.18 and 4.19 the number of node reloaded and reloaded alive are shown respectively , with respect to pause time. Figure 4.20 shows the packet delivery ratio during the simulation period, with respect to pause time.

Since a longer simulation time and heavier CBR traffic streams are chosen, nearly 90% to 98% percent of nodes have to be reloaded sometime during the simulation. (shown in Figure 4.18). The difference is: with PAWF, more than 90% to 94% nodes are reloaded before the battery completely runs out. In comparison, the ratio for AODV and GPSR is only 60% to 70% (shown in Figure 4.19). The reason for this result is that PAWF has the

ability to balance the usage of nodes based on the battery situation. Compared with AODV and GPSR, nodes with low energy are usually allocated light workload and live longer with PAWF. With this property, when the battery level is below the critical battery warning threshold (5000mWsec in this thesis), nodes with PAWF usually have enough time to be reloaded before turning off. Since most of the nodes are reloaded before dead with PAWF, the network partition could be avoided and packet drops are also reduced efficiently. With our experimental results shown in Figure 4.20, PAWF has 97% to 98% packet delivery ratio, 3% to 5% better than AODV and GPSR in a reloadable system during the simulation time. This is another benefit we obtain from the node over-usage avoidance of PAWF.

Compared with PAWF, LEAR_0.1 has a very similar delivery ratio and LEAR_0.4 has a slightly lower delivery ratio, but still higher than the other two non-power-aware approaches, GPSR and AODV. Like PAWF, the good delivery performance of LEAR comes from its ability to avoid the over-usage of low-battery nodes. With LEAR, low-battery nodes usually drop route requests and do not take part in data transmission work. For this reason, low-battery nodes usually keep alive for enough time to be reloaded before running out of battery. From Figure 4.19, the ratio of node reloaded alive by LEAR_0.1 is very high, about 90% to 93%. So LEAR_0.1, like PAWF, keeps network integrity very well in a reloadable network and then achieves a higher delivery ratio than GPSR and AODV. For LEAR_0.4, its power-awareness ability is weakened by a relative large adjustment factor d. So its has lower delivery ratio than LEAR_0.1 and PAWF.

4.6 Chapter Summary

This chapter presents the implementation of PAWF over GlomoSim and the comparison of performance with AODV, GPSR and LEAR. Experiments are designed and made under three energy scenarios: full energy, scarce energy and scarce energy with reloading. Based on the experimental results, PAWF shows its ability to balance the energy usage, prolong

the network life time and balance the workload of nodes. PAWF also shows that it can work well in high mobility networks and keep steady routing performance under different traffic loads. With pure position-based local routing, PAWF achieves energy balance of nodes without any extra routing control messages.

Chapter 5

Discussion

In this chapter, we describe the characteristics of networks and the energy consumption situation where PAWF is expected to work best and worst. Moreover we discuss the system design choices for PAWF, including the effect of the neighbor table update strategy, the usage of location services and possible improvements of PAWF.

5.1 The PAWF Properties with Different Network Characteristics

The simulation results presented in Chapter 4 show that PAWF efficiently balances the energy usage of nodes and significantly increases the network lifetime in MANETs. The power-awareness of PAWF is affected by the network density. When the network is dense, every local sender has a relatively bigger neighbor table and larger candidate set. So PAWF has more choices for the routing decision at every hop and higher probability to find an intended next hop node with high forwarding achievement and good battery situation. When the network becomes sparse, the size of neighbor table of every node is reduced and the number of possible pathes from the source to the destination is also decreased. So PAWF has few choices in local routing and the benefit of PAWF becomes limited. Moreover, when the network becomes sparse, the power-aware greedy forwarding process has a relative high probability to fail and PAWF works more frequently in the recovery mode. With

the usage of the planar graph face-crossing method, the routing decisions in the recovery mode are made in a non-power-awareness way. Therefore, the power-awareness of PAWF could be weakened with the decrease of the network density. How to increase the power-awareness in the sparse network is a very interesting topic. Possible ideas include adding power-awareness into the recovery mode or improving the current routing-mode switching strategy. Details of these ideas are presented in Section 6.2 as the future work.

Unlike topology-based routing algorithms, PAWF does not use end-to-end routing queries in forwarding. The chance that PAWF successfully transmits a packet at each hop, whether in the greedy mode or recovery mode, only depends on the local information of the current node and its strict neighbors. The accuracy of such local information is independent on the whole topology of the network. Therefore, the routing accuracy of PAWF is independent from the network scale. With the size of the network increasing, the performance of PAWF keeps steady and robust, which is an advantage, compared with topology-based algorithms.

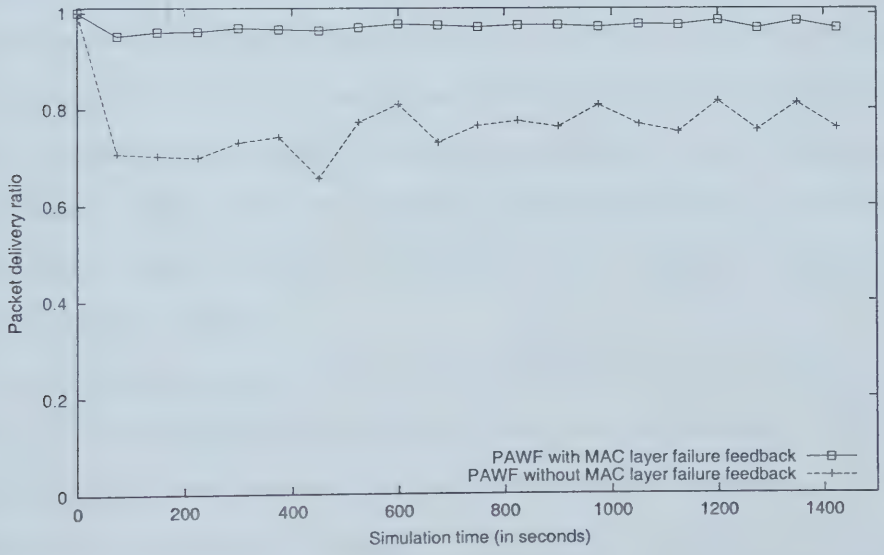


Figure 5.1: The routing performance of PAWF with/without MAC layer failure feedback: 300 CBRs, CBR interval = 0.2s/packet, pause time = 0s, simulation time = 1500s

The performance of position-based routing could be affected by the node mobility. It

is possible that nodes move out or move into the communication range of the current node during the packet transmission. In this case, packets could be delivered to the fake neighbor and routing fails due to the stale neighbor table information. Therefore, the update of neighbor tables is important for position-based routing. That is why additional neighbor table update strategies are used in PAWF. These strategies are very helpful to keep the neighbor table fresh and reduce the delivery failures. For example, the MAC layer failure feedback method is very efficient to avoid continuous delivery failures induced by stale neighbor table entries. When a neighbor is chosen as the intended next hop node and coincidentally moving out of the current node radio range, the delivery of the first packet to that node fails due to the physical disconnection. In this situation, the MAC lay failure feedback triggers a neighbor table update in the network layer and the corresponding stale neighbor entry is deleted from the neighbor table. Therefore, the following packets will not be sent to that fake neighbor again and continuous delivery failures are avoided. The experimental results of PAWF routing with/without MAC layer feedbacks is shown in Figure 5.1, where the pause time of mobile nodes is equal to 0 second. So, every node moves continuously and operates under high mobility. From the chart, it shows that PAWF with a MAC layer failure feedback strategy has about 10 % higher packet delivery ratio than that without MAC layer failure feedbacks. Therefore, the additional neighbor table update strategies are good complements of regular neighbor table updates, which ensure the robust routing of PAWF in networks with mobility.

Based on the discussion in Section 3.4.5, PAWF is loop free and guarantees to reach 100% of connected destinations in static networks. When the nodes are mobile, such a routing property can not be ensured completely due to the topology changes. For example, based on the discussion in [2], an inner-face loop problem could happen in the recovery process when nodes are movable. This inner face may not cross the line between the entrance point of the current recovery process and the packet's destination. In this case, it

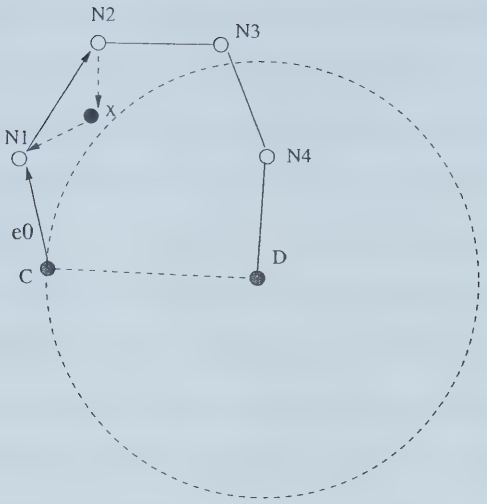


Figure 5.2: Inner-face problem: endless loop within the inner face

will induce loop around the inner face. An example of the inner-face problem appears in Figure 5.2(from the paper [2]), where the node C sends packets to the node D. The packet forwarding falls into the recovery mode at node C and traverses the face by the right-hand rule. The hop $hop0 = (C, N1)$ is the first hop of the current face crossed. When the packet is forwarded from $N1$ to $N2$, a node x moves such that the two dashed edges are newly created and a new inner face $(N1, N2, x)$ is generated. In the following, the packet is forwarded to node x, and then $N1$ based on the right-hand rule and a loop is formed. Since the new inner face is not crossed by the line $\overline{CD}$, no face-changing could be made during the transmission. Also because the first hop $hop0$ is not included by the inner face, $hop0$ is not useful in detecting this inner loop. Therefore, the packet will loop within the inner face forever or until the TTL of the packet is decremented to zero.

The inner-face problem in the recovery process is induced by the node mobility. But mobility may also enable nodes to get over this problem. For example, when the packet loops within the inner face $(N1, N2, x)$, if the node x or $N2$ moves out of the radio range of $N1$, the loop will be broken and packet forwarding returns normal. Moreover, Brad Nelson

Karp [2] discusses that the inner-face problem occurs less frequently as density decreases. *The closing of an inner face requires nodes to be placed to form an entire face within the outer one, which requires relatively more nodes lying around. But recovery routing is used more often as density decreases, or in the sparse network. So the probability of the occurrence of the inner-face problem is low* [2]. Based on our research, the inner-face problem is naturally reduced by the implementation of PAWF. In PAWF, the periodical beacon is the main method to maintain the neighbor table. When the node x moves into the radio range of node $N1$ and $N2$, the new neighbor entry of x usually can not be created or updated in $N1$ and $N2$ immediately. The create or update of the neighbor x has to wait until a beacon message or piggy-back information comes from x. Such a waiting period is usually longer than the packet forwarding delay from $N1$ to $N2$. In this case, the packet will still be forwarded from $N2$ to $N3$ with the original neighbor table and then reaches the destination successfully, even if x exists in the radio range of $N2$. Based on the above discussion, PAWF is sufficiently robust to be used and contribute to good packet delivery ratio in mobility networks.

In summary, PAWF forwards data packets only based on the local neighbor information. Its forwarding routing overhead is not affected by the node mobility and traffic load. So PAWF is scalable and works well in large-scale networks. PAWF is proven loop-free in the static network, and work robustly when nodes are movable. The PAWF's ability to balance the energy usage of nodes decreases with the decrease of the network density. So possible improvement could be done in the routing recovery process to increase the power-awareness of PAWF in the sparse network.

5.2 PAWF vs. Energy Consumption

In MANETs, PAWF can not control which node to be the packet source sender or final receiver, but it can choose which to be the next sender or relaying node. The ability of

PAWF to balance the energy consumption of mobile nodes comes from its ability to allocate transmission workload based on the energy consumption situation. If a candidate has consumed much more energy than other candidates, it is usually not chosen as the relaying node of the data traffic. Since those battery over-used nodes are usually avoided from packet forwarding, they are only allocated with light workload until they have a similar battery level to other nodes again. Therefore, the power-awareness of PAWF is affected by the proportion of energy consumed by nodes. If the energy consumed in the transmission and receiving dominates the whole energy consumption of nodes, PAWF works best. If the main part of energy were consumed in idling (by overhearing), the ability of PAWF would be reduced, because the part of energy consumption adjusted by PAWF is only a small part of the whole and does not affect the overall energy consumption of nodes greatly.

In Section 3.2, we have introduced several approaches to reduce the energy consumed in overhearing of nodes. The main idea of these approaches is to turn the radio off when possible. Radio-off can be controlled based on the information from the MAC layer or upper layers. For example, in the Adaptive Fidelity Energy-Conserving Algorithm (AFECA), the information from above the MAC layer is employed to control the radio power. Different from AFECA, the Power-Aware Multi-Access protocol with Signalling (PAMAS) protocol is built up over the MAC layer. With the suitable radio-off method, energy consumed in idling is reduced greatly. If the MAC layer and above-MAC layer radio-off strategies are combined, more energy-saving is expected.

Furthermore, we know, in a base station environment (such as in a GSM system), the resource-rich base station moderates communication among nodes, scheduling and buffering traffic to reduce contention and allowing more mobile devices to spend as much time as possible in a low-energy-consumption sleep mode. Therefore, if we emulate some of the energy efficiency associated with the base station mode and design a hybrid protocol, the energy consumption in idle time can be further reduced. For example, we can partition

the whole network into small areas. Within every area, one or several representatives are chosen to buffer traffic of the local area. Within the local area, a base-station mode routing is used and the route from the source to destination is decided by the representative node. Among different areas, the regular Ad Hoc routing or cluster-based routing could be chosen. The difficulty of this approach is the dynamic of the cluster structure and the complexity of representative election and cluster maintenance. Despite these issues, modification of existing clustering routing algorithms such as [46] to support such a hybrid energy-efficiency strategy may prove worthwhile [47].

In summary, PAWF balances the energy consumption of nodes based on the difference of their battery status. Currently, PAWF can only control the energy consumption in the packet transmission and corresponding packet receiving. If the energy consumed during the idle time dominates the whole energy consumption of nodes, the ability of PAWF for power-awareness is limited. Based on existing research results, with a suitable radio-off method, it is possible that the energy consumed during transmitting and receiving becomes the main part of the whole energy consumption, which ensures PAWF's ability to achieve energy consumption balance among the nodes.

5.3 The Effect of Location Services

Besides the strict neighbor information, PAWF routing also requires the location information of the packet destination. Such information is used to calculate the forwarding achievement or choose the next edge of face-crossing in the recovery mode. In some applications, the destination positions may be known or predictable by the source sender. For example, for the Internet service, the Internet access server should be fixed and accessed by all the mobile nodes. In the Public Transit System, the routes of the buses are fixed. So we can predict the location of the bus based on its departure time and average velocity. In the above cases, no location services are needed.

If the position of the destination node is not fixed or can not be extrapolated, the help of a location service is needed. When a node does not know the destination position of a packet, it contacts the location service and requests that information. It is important to note that only the packet source sender needs to query the location service for the destination information and that information will be used without change by the following nodes through the path.

The addition of location update and lookup will increase PAWF's routing overhead. Based on the building up and updating strategy, location services can be divided into two groups: centralized and distributed.

The centralized location service asks for some specific nodes to work as the location servers, each maintaining position information of all nodes. These location servers are similar to the base station of regular cellar phone systems. In MANETs, such a centralized approach is viable only as an external service that can be reached via a non-Ad-Hoc method, because it is difficult to obtain the position of the location servers if the servers were part of the Ad-Hoc network [15]. Moreover, we can not guarantee that every node is physically connected to at least one location server in a dynamic mobile Ad Hoc network.

In a distributed location service, the straightforward approach to build up the location database is broadcast. This method is used in DREAM [14], where each mobile node maintains a position database that stores position information about every mobile node in the network. Each node periodically broadcasts a control packet containing its own position. A node can control the accuracy of its position information available to other nodes by (a) the frequency of broadcasting; (b) how far a position update may travel before it is discarded. This is a method offering a relatively reliable location service, but heavy overhead at the same time.

In order to reduce the overhead, two issues are important for the location service: hierarchical structure and combining with the regular packet routing. With such requirements,

Grid Location Service (GLS) [33] is a choice. GLS geographically divides the network area into a hierarchy of squares. In this hierarchy, a *n-order* square contains exactly four *(n-1)order* squares, forming a quad-tree of squares, in which a particular *(n-1)order* square is part of only one *n-order* square and no overlap happens. Each node periodically updates a small set of other nodes (its location servers) with its current location. The selection of local servers of the node is assisted by a predefined ordering of node identifiers. Queries for a mobile node's location also use the predefined identifier ordering and spatial hierarchy to find a location server for that node. With the hierarchical structure, every node only needs to update its location to a small set of nodes. So the overhead of location service is reduced. Moreover, both the queries and updates of the location database in GLS can be achieved based on pure position-based routing, such as PAWF, not broadcast flooding, which ensures the pure position-based routing structure, so as to keep the low routing overhead and the scalability of the protocol.

In conclusion, an accurate location service is important for position-based routing without fixed or predictable destinations. The overhead of the location service will increase the total overhead of the routing protocol. Based on the dynamic property of MANETs, a distributed location service is more applicable than a centralized one. Building up the distributed location databases with a hierarchical structure is helpful to reduce the overhead of location service.

5.4 Chapter Summary

In this chapter, We present the effect of the network property to the performance of PAWF. Since PAWF is a local-routing algorithm, it is scalable and works well in large-scale networks. The PAWF's ability to balance the energy usage of nodes is relative to the network density and energy consumption situation of nodes. If the network is sparse or energy consumed in idle time becomes crucial, PAWF's ability of power-awareness is weakened. A

good location service is important for PAWF when the destinations are not fixed or non-predictable. With analysis, we show that a distributed location service with a hierarchical structure is suitable for PAWF and other pure position-based routing algorithms.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

We have presented PAWF, a power-aware position-based routing protocol. With the usage of an asymmetrical exponential weighted forwarding function, our protocol avoids the over-usage of mobile nodes. Therefore, PAWF can increases the lifetime of the network and packet delivery ratio correspondingly. Such improvement is held under different node mobility levels and network workloads. From simulation results, we find PAWF has low and constant forwarding routing overhead, which is not affected by the mobility of nodes and network traffic. So PAWF is more energy-efficient and scalable than regular topology-based routing approaches, such as AODV. PAWF is loop-free in the static network and keeps the routing efficiency like regular position-based approaches, such as GPSR. PAWF achieves energy-balance of nodes only based on the local information of neighbor and no extra routing control messages are needed, which is an advantage, comparing with topology-based power-aware routing protocols, such as LEAR.

We note that the performance of PAWF is affected by the efficiency and the accuracy of the location service and the proportion of the energy consumed in different operation modes (transmitting, receiving, sleeping or idling). The ability of PAWF to balance the energy usage of nodes is also relative to the density of the network. In the sparse network, such ability is weakened with the decrease of the number of local choices. PAWF is robust

and works well in networks with mobility, but it does not guarantee the loop-free routing when the mobility of nodes is high.

Based on the research in this thesis, we achieve:

- The principle and structure of the position-based routing process, the description of implementing position-based routing with a local graph and the definition of the weighted forwarding function. With the usage of weighted forwarding function, the localized power-awareness routing and further QoS applications become possible.
- The design and application of an asymmetrical exponential weighted forwarding function, which combines the energy consumed value, energy residual value with the forwarding achievement.
- PAWF, a pure position-based power-aware routing algorithm, which performs all forwarding decisions at the current local node using only information concerning that node's strict neighbors and the destination.
- The full PAWF routing protocol designed for MANETs, including greedy power-aware routing with a weighted forwarding function, existing planar graph recovery, neighbor tables building up with periodical beaconing, MAC layer failure feedback and piggybacking neighbor updates.
- A simulation-based performance evaluation of PAWF, AODV, GPSR and LEAR on three energy scenarios with different mobility patterns and network loads, that shows PAWF balances the energy usage of nodes and significantly increases the network lifetime.
- The analysis of the network characteristics and energy consumption characteristics where PAWF is expected to work best and worst. At the same time, the existing location services and the availability of implementing them in existing position-based routing algorithms are also discussed.

6.2 Future Directions for Work

The first possible research topic is to design a hybrid routing message exchange strategy, which makes the tradeoff between routing performance and routing cost. One possible idea is to combine regular periodical beacons with on-demand unicast beacons. With this strategy, PAWF is expected to perform more accurately in the high mobility networks, while keeping low routing overhead.

Furthermore, in order to increase the power-awareness ability of PAWF, we need to keep PAWF working in a power-aware mode as much as possible. Two possible improvements of PAWF are available:

- To design a new switching strategy to force PAWF to return to the power-aware greedy mode sooner. So PAWF works longer in the power-aware greedy mode
- To design an energy-balanced face-crossing algorithm to further optimize the energy consumption of mobile nodes in the routing recovery mode.

In addition, some relative topics of PAWF are also worthwhile to be studied to ensure PAWF's ability to balance the energy consumption of nodes. These topics include:

- To combine the MAC layer and above-MAC layer radio-off strategies to further reduce the energy consumption of nodes in idling.
- To design an energy-balanced radio-off strategy to deeply balance the energy consumption of nodes.
- To design a hybrid structure of the network by combining the pure Ad Hoc network with the base-station network structure so as to centralize the traffic control, increase the routing efficiency and optimize the energy consumption.

Moreover, since PAWF is able to find global power-aware near-optimal paths by local step by step forwarding, it is also a good foundation above which to implement further QoS

routing, including workload balance, real-time communication and bandwidth-constraint services. Such routing protocols should allow more efficient routing, longer network life-time, higher scalability and more adaptive bandwidth reservation/allocation.

Bibliography

[1] J. P. Macker and M. S. Corson. Mobile ad hoc networking and the IETF. *Computing and Communications Review*, 2, 1:9–14, 1998.

[2] Brad Karp and H. T. Kung. GPSR: Greedy perimeterstateless routing for wireless networks. In *ACM MobiCom*, Aug. 2000.

[3] Xiang Zeng, Rajive Bagrodia, and Mario Gerla. GloMoSim: a library for parallel simulation of large-scale wireless networks. In *Proceedings of the 12th Workshop on Parallel and Distributed Simulations – PADS '98*, Banff, Alberta, Canada, May 1998.

[4] C. E. Perkins, E. M. Belding-Royer, and S. R. Das. Ad hoc on-demand distance vector (AODV) routing, IETF. *Internet Draft (Work in Progress)*, June 2002.

[5] Kyungtae Woo, Chansu Yu, Dongman Lee, Hee Yong Youn, and Lee B. Non-blocking, localized routing algorithm for balanced energy consumptionin mobile ad hoc networks. In *MASCOTS*, pages 117–124, Cincinnati, OH, USA, Aug. 2001.

[6] IEEE. part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. *IEEE Standard 802.11*, 1999.

[7] P. Karn. Maca - a new channel access method for packet radio. In *ARRL/CRRL Amateur Radio 9th Computer Networking Conference*, pages 134–140, 1990.

[8] C. E. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *SIGCOMM Symposium on Communications Architectures and Protocols*, pages 212–225, Sept. 1994.

[9] C.-C. Chiang. Routing in clustered multihop, mobile wireless networks with fading channel. In *IEEE SICON '97*, pages 197–211, Apr. 1997.

[10] S. Murthy and J. J. Garcia-Luna-Aceves. An efficient routing protocol for wireless networks. *ACM Mobile Networks and App. J., Special Issue on Routing in Mobile Communication Networks*, pages 183–197, Oct. 1996.

[11] D. B. Johnson and D. A. Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*, Mobile Computing, Imielinski and Korth, Eds. Kluwer Academic Publishers, 1996.

[12] V. Park and S. Corson. Temporally-ordered routing algorithm (TORA). *version 1: Functional Specification. IETF Internet draft*, July 2001.

[13] Young-Bae Ko and Nitin H. Vaidya. Location-aided routing (LAR) in mobile ad hoc networks. In *IEEE MobiCom*, 1998.

[14] S. Basagni, I. Chlamtac, V. R. Syrotiuk, and B. A. Woodward. A distance routing effect algorithm for mobility (DREAM). In *Fourth Annual ACM/IEEE International Conference in Mobile Computing and Networking (MobiCom)*, 1998.

[15] M. Mauve, J. Widmer, and H. Hartenstein. A survey on position-based routing in mobile ad hoc networks. *IEEE Network Magazine*, 15(6):30–39, Nov. 2001.

[16] I. Stojmenovic and X. Lin. GEDIR: Loop-free location based routing in wireless networks. In *IASTED Int. Conf on Parallel and Distributed Computing and Systems*, Boston, MA, USA, Nov 1999.

[17] P. Bergamo, D. Maniezzo, A. Travasoni, A. Giovanardi, G. Mazzini, and M. Zorzi. Distributed power control for energy efficient routing in ad hoc networks. In *European Wireless 2002*, pages 237–243, Florence, Italy, Feb. 2002.

[18] E.W. Dijkstra. A note on two problems in connexion with graphs. *Numer. Math.*, 1:269–271, Oct. 1959.

[19] J. Gomez, A. Campbell, M. Naghshineh, and C. Bisdikian. Conserving transmission power in wireless ad hoc networks. In *International Conference on Networking Protocols*, Nov 2001.

[20] V. Rodoplu and T. H. Meng. Minimum energy mobile wireless networks. *IEEE J. Selected Areas in Comm.*, 17(8):1333–1344, Aug. 1999.

[21] Ivan Stojmenovic and Xu Lin. Power-aware localized routing in wireless networks. In *IEEE Int. Parallel and Distributed Processing Symp.*, 2000.

[22] Yuan Xue and Baochun Li. A location-aided power-aware routing protocol in mobile ad hoc networks. In *IEEE Globecom 2001*, volume Vol. 5, pages 2837–2841, San Antonio, Texas, Nov. 2001.

[23] I. Batsiolas and Ioanis Nikolaidis. Selective idling: Experiments in transport layer energy conservation. *The Journal of Supercomputing*, 20(2):101–114, 2001.

[24] Y. Xu, J. Heidemann, and D. Estrin. Adaptive energy-conserving routing for multihop ad hoc networks. In *Technical Report TR-2000.*

[25] Ya Xu, John Heidemann, and Deborah Estrin. Geography-informed energy conservation for ad hoc routing. In *the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 70–83, Rome, Italy, July 2001.

[26] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. Span: An energyefficient coordination algorithm for topology maintenance in ad hoc wireless networks. In *ACM/IEEE MobiCom*, 2001.

[27] S. Singh and C. S. Raghavendra. Pamas-power aware multi-access protocol with signaling for ad hoc networks. *ACM Communications Review*, July 1998.

[28] J. Aslam Q. Li and D. Rus. Online power-aware routing in wireless ad-hoc networks. In *the Seventh Annual International Conference on Mobile Computing and Networking*, pages 97–107, July 2001.

[29] J.H. Chang and L. Tassiulas. Energy conserving routing in wireless ad-hoc networks. In *IEEE INFOCOM*, 2000.

[30] A. Misra and S. Banerjee. Mrpc: Maximizing network lifetime for reliable routing in wireless environments. In *IEEE Wireless Communications and Networking Conference (WCNC)*, Mar. 2002.

[31] O.Kassten. Energy consumption. *http://www.inf.ethz.ch/ kasten/research/bathtub/energy_consumption.html*, 2001.

[32] Z. J. Haas and B. Liang. Ad hoc mobility management with uniform quorum systems. *IEEE/ACM Trans. on Networking*, 7(2):228–240, April 1999.

[33] J. Li, J. Jannotti, D. De Couto, D. Karger, and R. Morris. A scalable location service for geographic ad hoc routing. In *6th ACM Intl. Conf. on Mobile Computing and Networking*, pages 120–130, Boston, Massachusetts, Aug. 2000.

[34] H. Takagi and L. Kleinrock. Optimal transmission ranges for randomly distributed packet radio terminals. *IEEE Trans. on Comm.*, 32(3):246–257, March 1984.

[35] T.-C. Hou and V. O.K. Li. Transmission range control in multihop packet radio networks. *IEEE Trans. on Communications*, 34(1):38–44, Jan. 1986.

[36] H. Singh E. Kranakis and J. Urrutia. Compass routing on geometric networks. In *11th Canadian Conf. on Computational Geometry*, Vancouver, August 1999.

[37] Ivan Stojmenovic and Mark Russell. Depth first search and location based localized routing and qos routing in wireless networks. In *IEEE International Conference on Parallel Processing*, pages 173 –180,, Aug. 2000.

[38] Clifford W. Marshall. Applied graph theory. *John Wiley and Sons*, Jan. 1971.

[39] National institute of standards and technology (NIST). *http://www.nist.gov/dads/HTML/planarstrght.html*.

[40] K. Gabriel and R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Zoology*, 18:259–278, 1969.

[41] John Day. The (un)revised osi reference model. *Computer Communication Review, ACM SIGCOMM*, Oct. 1995.

[42] T. S. Rappaport. Wireless communications. *Principles and Practice*.

[43] Jon Postel. Internet protocol. *RFC 791, ISI*, September 1981.

[44] Hosh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, and Jorieta Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Mobicom98*, 1998.

[45] Cucent Technologies. Wavelan IEEE 802.11 PC card user's guide. Feb. 1999.

[46] M.Jiang, J.Li, and Y.C.Tay. Cluster based routing protocol (CBRP) function specifications. *IETF Draft*, Aug 1999.

[47] L. M. Feeney and M. Nilsson. Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In *INFOCOM 2001*, Anchorage, AK, US, April 2001.

University of Alberta Library
0 1620 1748 6604
B45822